S-Finite Measure Monad on Quasi-Borel Spaces

Michikazu Hirata, Yasuhiko Minamide

March 19, 2025

Abstract

The s-finite measure monad on quasi-Borel spaces provides a suitable denotational model for higher-order probabilistic programs with conditioning. This entry is a formalization of the s-finite measure monad and related notions, including s-finite measures, s-finite kernels, and a proof automation for quasi-Borel spaces which is an extension of our previous entry *quasi-Borel spaces*. We also implement several examples of probabilistic programs in previous works and prove their property.

This work is a part of the work by Hirata, Minamide, and Sato, *Semantic Foundations of Higher-Order Probabilistic Programs in Isabelle/HOL* which will be presented at the 14th Conference on Interactive Theorem Proving (ITP2023).

Contents

For the terminology of s-finite measures/kernels, we refer to the work by Staton [4]. For the definition of the s-finite measure monad, we refer to the lecture note by Yang [6]. The construction of the s-finite measure monad is based on the detailed pencil-and-paper proof by Tetsuya Sato.

1 Lemmas

theory *Lemmas-S-Finite-Measure-Monad*
  imports *HOL−Probability.Probability Standard-Borel-Spaces.StandardBorel*
begin

lemma *integrable-mono-measure*:
  fixes f :: $'a \Rightarrow 'b$::{*banach*, *second-countable-topology*}
  assumes [*measurable-cong*,*measurable*]:*sets M* = *sets N M* ≤ *N integrable N f*
  shows *integrable M f*
  ⟨*proof*⟩

lemma *AE-mono-measure*:
  assumes *sets M* = *sets N M* ≤ *N AE x in N*. *P x*
  shows *AE x in M*. *P x*
  ⟨*proof*⟩

lemma *finite-measure-return*:*finite-measure* (*return M x*)
  ⟨*proof*⟩

lemma *nn-integral-return′*:
  assumes $x \notin$ *space M*
  shows ($\int^+$ *x*. *g x* ∂*return M x*) = *0*
⟨*proof*⟩

lemma *pair-measure-return*: *return M l* $\bigotimes_M$ *return N r* = *return* (*M* $\bigotimes_M$ *N*) (*l*,*r*)
⟨*proof*⟩

lemma *null-measure-distr*: *distr* (*null-measure M*) *N f* = *null-measure N*
⟨*proof*⟩

lemma *integral-measurable-subprob-algebra2*:
fixes *f* :: - ⇒ - ⇒ -::{*banach*,*second-countable-topology*}
assumes [*measurable*]: (λ(*x*, *y*). *f x y*) ∈ *borel-measurable* (*M* $\bigotimes_M$ *N*) *L* ∈ *measurable M* (*subprob-algebra N*)
shows (λ*x*. *integral*^{L} (*L x*) (*f x*)) ∈ *borel-measurable M*
⟨*proof*⟩

lemma *distr-id'*:
assumes *sets N* = *sets M*
and ⋀*x*. *x* ∈ *space N* ⟹ *f x* = *x*
shows *distr N M f* = *N*
⟨*proof*⟩

lemma *measure-density-times*:
assumes [*measurable*]:*S* ∈ *sets M X* ∈ *sets M r* ≠ ∞
shows *measure* (*density M* (λ*x*. *indicator S x* ∗ *r*)) *X* = *enn2real r* ∗ *measure M* (*S* ∩ *X*)
⟨*proof*⟩

lemma *complete-the-square*:
fixes *a b c x* :: *real*
assumes *a* ≠ *0*
shows $a{*}x^2 + b * x + c = a * (x + (b\ /\ (2{*}a)))^2 - ((b^2 - 4{*}\ a * c)/(4{*}a))$
⟨*proof*⟩

lemma *complete-the-square2'*:
fixes *a b c x* :: *real*
assumes *a* ≠ *0*
shows $a{*}x^2 - 2 * b * x + c = a * (x - (b\ /\ a))^2 - ((b^2 - a{*}c)/a)$
⟨*proof*⟩

lemma *normal-density-mu-x-swap*:
normal-density μ σ *x* = *normal-density x* σ μ
⟨*proof*⟩

lemma *normal-density-plus-shift*: *normal-density* μ σ (*x* + *y*) = *normal-density* (μ − *x*) σ *y*
⟨*proof*⟩

lemma *normal-density-times*:
assumes σ > *0* σ' > *0*
shows *normal-density* μ σ *x* ∗ *normal-density* μ' σ' *x* = (*1* / *sqrt* (*2* ∗ *pi* ∗

$(\sigma^2 + \sigma'^2))) * exp\ (- (\mu - \mu')^2 / (2 * (\sigma^2 + \sigma'^2))) * normal\text{-}density\ ((\mu*\sigma'^2 + \mu'*\sigma^2)/(\sigma^2 + \sigma'^2))\ (\sigma * \sigma' / sqrt\ (\sigma^2 + \sigma'^2))\ x$
(**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma *KL-normal-density*:
assumes [*arith*]: $b > 0$ $d > 0$
shows *KL-divergence* (*exp 1*) (*density lborel* (*normal-density a b*)) (*density lborel* (*normal-density c d*)) $= ln\ (b / d) + (d^2 + (c - a)^2) / (2 * b^2) - 1 / 2$ (**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma *count-space-prod*:*count-space* (*UNIV* :: (*'a* :: *countable*) *set*) $\bigotimes_M$ *count-space* (*UNIV* :: (*'b* :: *countable*) *set*) = *count-space UNIV*
⟨*proof*⟩

lemma *measure-pair-pmf*:
fixes *p* :: (*'a* :: *countable*) *pmf* **and** *q* :: (*'b* :: *countable*) *pmf*
shows *measure-pmf p* $\bigotimes_M$ *measure-pmf q* = *measure-pmf* (*pair-pmf p q*) (**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma *distr-PiM-distr*:
assumes *finite I* $\bigwedge i.\ i \in I \Longrightarrow$ *sigma-finite-measure* (*distr* (*M i*) (*N i*) (*f i*))
and $\bigwedge i.\ i \in I \Longrightarrow f\ i \in M\ i \rightarrow_M N\ i$
shows *distr* $(\Pi_M\ i{\in}I.\ M\ i)\ (\Pi_M\ i{\in}I.\ N\ i)\ (\lambda xi.\ \lambda i{\in}I.\ f\ i\ (xi\ i)) = (\Pi_M\ i{\in}I.$ *distr* (*M i*) (*N i*) (*f i*))
⟨*proof*⟩

lemma *distr-PiM-distr-prob*:
assumes $\bigwedge i.\ i \in I \Longrightarrow$ *prob-space* (*M i*)
and $\bigwedge i.\ i \in I \Longrightarrow f\ i \in M\ i \rightarrow_M N\ i$
shows *distr* $(\Pi_M\ i{\in}I.\ M\ i)\ (\Pi_M\ i{\in}I.\ N\ i)\ (\lambda xi.\ \lambda i{\in}I.\ f\ i\ (xi\ i)) = (\Pi_M\ i{\in}I.$ *distr* (*M i*) (*N i*) (*f i*))
⟨*proof*⟩

end

2 Kernels

theory *Kernels*
imports *Lemmas-S-Finite-Measure-Monad*
begin

2.1 S-Finite Measures

locale *s-finite-measure* =
fixes *M* :: *'a measure*
assumes *s-finite-sum*: $\exists\,Mi :: nat \Rightarrow$ *'a measure*. $(\forall\,i.\ sets\ (Mi\ i) = sets\ M) \wedge$

$(\forall\, i.\ \textit{finite-measure}\ (Mi\ i)) \wedge (\forall\, A{\in}\textit{sets}\ M.\ M\ A = (\sum i.\ Mi\ i\ A))$

lemma(**in** *sigma-finite-measure*) *s-finite-measure*: *s-finite-measure M*
⟨*proof*⟩

lemmas(**in** *finite-measure*) *s-finite-measure-finite-measure* = *s-finite-measure*

lemmas(**in** *subprob-space*) *s-finite-measure-subprob* = *s-finite-measure*

lemmas(**in** *prob-space*) *s-finite-measure-prob* = *s-finite-measure*

sublocale *sigma-finite-measure* ⊆ *s-finite-measure*
⟨*proof*⟩

lemma *s-finite-measureI*:
assumes $\bigwedge i.\ \textit{sets}\ (Mi\ i) = \textit{sets}\ M\ \bigwedge i.\ \textit{finite-measure}\ (Mi\ i)\ \bigwedge A.\ A{\in}\textit{sets}\ M \Longrightarrow M\ A = (\sum i.\ Mi\ i\ A)$
shows *s-finite-measure M*
⟨*proof*⟩

lemma *s-finite-measure-prodI*:
assumes $\bigwedge i\ j.\ \textit{sets}\ (Mij\ i\ j) = \textit{sets}\ M\ \bigwedge i\ j.\ Mij\ i\ j\ (\textit{space}\ M) < \infty\ \bigwedge A.\ A \in \textit{sets}\ M \Longrightarrow M\ A = (\sum i.\ (\sum j.\ Mij\ i\ j\ A))$
shows *s-finite-measure M*
⟨*proof*⟩

corollary *s-finite-measure-s-finite-sumI*:
assumes $\bigwedge i.\ \textit{sets}\ (Mi\ i) = \textit{sets}\ M\ \bigwedge i.\ \textit{s-finite-measure}\ (Mi\ i)\ \bigwedge A.\ A \in \textit{sets}\ M \Longrightarrow M\ A = (\sum i.\ Mi\ i\ A)$
shows *s-finite-measure M*
⟨*proof*⟩

lemma *s-finite-measure-finite-sumI*:
assumes $\textit{finite}\ I\ \bigwedge i.\ i \in I \Longrightarrow \textit{s-finite-measure}\ (Mi\ i)\ \bigwedge i.\ i \in I \Longrightarrow \textit{sets}\ (Mi\ i) = \textit{sets}\ M$
and $\bigwedge A.\ A \in \textit{sets}\ M \Longrightarrow M\ A = (\sum i{\in}I.\ Mi\ i\ A)$
shows *s-finite-measure M*
⟨*proof*⟩

lemma *countable-space-s-finite-measure*:
assumes *countable* (*space M*) *sets M* = *Pow* (*space M*)
shows *s-finite-measure M*
⟨*proof*⟩

lemma *s-finite-measure-subprob-space*:
$\textit{s-finite-measure}\ M \longleftrightarrow (\exists\, Mi :: \textit{nat} \Rightarrow \textit{'a measure}.\ (\forall\, i.\ \textit{sets}\ (Mi\ i) = \textit{sets}\ M) \wedge (\forall\, i.\ (Mi\ i)\ (\textit{space}\ M) \le 1) \wedge (\forall\, A{\in}\textit{sets}\ M.\ M\ A = (\sum i.\ Mi\ i\ A)))$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *finite-measures*:
obtains *Mi* **where** $\bigwedge i.\ sets\ (Mi\ i) = sets\ M$ $\bigwedge i.\ (Mi\ i)\ (space\ M) \le 1$ $\bigwedge A.\ M\ A = (\sum i.\ Mi\ i\ A)$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *finite-measures-ne*:
assumes $space\ M \neq \{\}$
obtains *Mi* **where** $\bigwedge i.\ sets\ (Mi\ i) = sets\ M$ $\bigwedge i.\ subprob\text{-}space\ (Mi\ i)$ $\bigwedge A.\ M\ A = (\sum i.\ Mi\ i\ A)$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *finite-measures'*:
obtains *Mi* **where** $\bigwedge i.\ sets\ (Mi\ i) = sets\ M$ $\bigwedge i.\ finite\text{-}measure\ (Mi\ i)$ $\bigwedge A.\ M\ A = (\sum i.\ Mi\ i\ A)$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *s-finite-measure-distr*:
assumes $f[measurable]$:$f \in M \to_M N$
shows *s-finite-measure* (*distr M N f*)
⟨*proof*⟩

lemma *nn-integral-measure-suminf*:
assumes [*measurable-cong*]:$\bigwedge i.\ sets\ (Mi\ i) = sets\ M$ **and** $\bigwedge A.\ A \in sets\ M \Longrightarrow M\ A = (\sum i.\ Mi\ i\ A)$ $f \in borel\text{-}measurable\ M$
shows $(\sum i.\ \int^+ x.\ f\ x\ \partial(Mi\ i)) = (\int^+ x.\ f\ x\ \partial M)$
⟨*proof*⟩

A *density M f* of *s*-finite measure M and $f \in$ *borel-measurable M* is again s-finite. We do not require additional assumption, unlike σ-finite measures.

lemma(**in** *s-finite-measure*) *s-finite-measure-density*:
assumes $f[measurable]$:$f \in borel\text{-}measurable\ M$
shows *s-finite-measure* (*density M f*)
⟨*proof*⟩

lemma
fixes $f :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$
assumes [*measurable-cong*]:$\bigwedge i.\ sets\ (Mi\ i) = sets\ M$ **and** $\bigwedge A.\ A \in sets\ M \Longrightarrow M\ A = (\sum i.\ Mi\ i\ A)$ *integrable M f*
shows *lebesgue-integral-measure-suminf*:$(\sum i.\ \int x.\ f\ x\ \partial(Mi\ i)) = (\int x.\ f\ x\ \partial M)$ (**is** *?suminf*)
and *lebesgue-integral-measure-suminf-summable-norm*: *summable* $(\lambda i.\ norm\ (\int x.\ f\ x\ \partial(Mi\ i)))$ (**is** *?summable2*)
and *lebesgue-integral-measure-suminf-summable-norm-in*: *summable* $(\lambda i.\ \int x.\ norm\ (f\ x)\ \partial(Mi\ i))$ (**is** *?summable*)
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *measurable-emeasure-Pair'*:
assumes $Q \in sets\ (N \bigotimes_M M)$

shows $(\lambda x.\ emeasure\ M\ (Pair\ x\ -`\ Q)) \in borel\text{-}measurable\ N$ (**is** *?s Q* ∈ -)
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *measurable-emeasure′*[*measurable* (*raw*)]:
assumes *space*: $\bigwedge x.\ x \in space\ N \Longrightarrow A\ x \subseteq space\ M$
assumes *A*: $\{x{\in}space\ (N \bigotimes_M M).\ snd\ x \in A\ (fst\ x)\} \in sets\ (N \bigotimes_M M)$
shows $(\lambda x.\ emeasure\ M\ (A\ x)) \in borel\text{-}measurable\ N$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *emeasure-pair-measure′*:
assumes $X \in sets\ (N \bigotimes_M M)$
shows $emeasure\ (N \bigotimes_M M)\ X = (\int^+ x.\ \int^+ y.\ indicator\ X\ (x, y)\ \partial M\ \partial N)$
(**is** - = *?μ X*)
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *emeasure-pair-measure-alt′*:
assumes *X*: $X \in sets\ (N \bigotimes_M M)$
shows $emeasure\ (N \bigotimes_M M)\ X = (\int^+ x.\ emeasure\ M\ (Pair\ x\ -`\ X)\ \partial N)$
⟨*proof*⟩

proposition (**in** *s-finite-measure*) *emeasure-pair-measure-Times′*:
assumes *A*: $A \in sets\ N$ **and** *B*: $B \in sets\ M$
shows $emeasure\ (N \bigotimes_M M)\ (A \times B) = emeasure\ N\ A * emeasure\ M\ B$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *measure-times*:
assumes[*measurable*]: $A \in sets\ N\ B \in sets\ M$
shows $measure\ (N \bigotimes_M M)\ (A \times B) = measure\ N\ A * measure\ M\ B$
⟨*proof*⟩

lemma *pair-measure-s-finite-measure-suminf*:
assumes *Mi*[*measurable-cong*]:$\bigwedge i.\ sets\ (Mi\ i) = sets\ M\ \bigwedge i.\ finite\text{-}measure\ (Mi\ i)\ \bigwedge A.\ M\ A = (\sum i.\ Mi\ i\ A)$
and *Ni*[*measurable-cong*]:$\bigwedge i.\ sets\ (Ni\ i) = sets\ N\ \bigwedge i.\ finite\text{-}measure\ (Ni\ i)\ \bigwedge A.\ N\ A = (\sum i.\ Ni\ i\ A)$
shows $(M \bigotimes_M N)\ A = (\sum i\ j.\ (Mi\ i \bigotimes_M Ni\ j)\ A)$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *pair-measure-s-finite-measure-suminf′*:
assumes *Mi*[*measurable-cong*]:$\bigwedge i.\ sets\ (Mi\ i) = sets\ M\ \bigwedge i.\ finite\text{-}measure\ (Mi\ i)\ \bigwedge A.\ M\ A = (\sum i.\ Mi\ i\ A)$
and *Ni*[*measurable-cong*]:$\bigwedge i.\ sets\ (Ni\ i) = sets\ N\ \bigwedge i.\ finite\text{-}measure\ (Ni\ i)\ \bigwedge A.\ N\ A = (\sum i.\ Ni\ i\ A)$
shows $(M \bigotimes_M N)\ A = (\sum i\ j.\ (Mi\ j \bigotimes_M Ni\ i)\ A)$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *pair-measure-s-finite-measure*:
assumes *s-finite-measure M* **and** *s-finite-measure N*

shows *s-finite-measure* $(M \bigotimes_M N)$
⟨*proof*⟩

lemma(in *s-finite-measure*) *borel-measurable-nn-integral-fst'*:
assumes [*measurable*]: $f \in$ *borel-measurable* $(N \bigotimes_M M)$
shows $(\lambda x. \int^+ y.\ f\ (x, y)\ \partial M) \in$ *borel-measurable* N
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *nn-integral-fst'*:
assumes f: $f \in$ *borel-measurable* $(M1 \bigotimes_M M)$
shows $(\int^+ x. \int^+ y.\ f\ (x, y)\ \partial M\ \partial M1) = integral^N\ (M1 \bigotimes_M M)\ f$ (**is** *?I f* =
-)
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *borel-measurable-nn-integral'*[*measurable* (*raw*)]:
case-prod $f \in$ *borel-measurable* $(N \bigotimes_M M) \Longrightarrow (\lambda x. \int^+ y.\ f\ x\ y\ \partial M) \in$
borel-measurable N
⟨*proof*⟩

lemma *distr-pair-swap-s-finite*:
assumes *s-finite-measure M1* **and** *s-finite-measure M2*
shows $M1 \bigotimes_M M2 = distr\ (M2 \bigotimes_M M1)\ (M1 \bigotimes_M M2)\ (\lambda(x, y).\ (y, x))$ (**is**
?P = *?D*)
⟨*proof*⟩

proposition *nn-integral-snd'*:
assumes *s-finite-measure M1 s-finite-measure M2*
and f[*measurable*]: $f \in$ *borel-measurable* $(M1 \bigotimes_M M2)$
shows $(\int^+ y.\ (\int^+ x.\ f\ (x, y)\ \partial M1)\ \partial M2) = integral^N\ (M1 \bigotimes_M M2)\ f$
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *borel-measurable-lebesgue-integrable'*[*measurable* (*raw*)]:
fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]: *case-prod* $f \in$ *borel-measurable* $(N \bigotimes_M M)$
shows *Measurable.pred N* (λ*x*. *integrable M* (*f x*))
⟨*proof*⟩

lemma (**in** *s-finite-measure*) *measurable-measure'*[*measurable* (*raw*)]:
$(\bigwedge x.\ x \in space\ N \Longrightarrow A\ x \subseteq space\ M) \Longrightarrow$
$\{x \in space\ (N \bigotimes_M M).\ snd\ x \in A\ (fst\ x)\} \in sets\ (N \bigotimes_M M) \Longrightarrow$
$(\lambda x.\ measure\ M\ (A\ x)) \in$ *borel-measurable* N
⟨*proof*⟩

proposition (**in** *s-finite-measure*) *borel-measurable-lebesgue-integral'*[*measurable* (*raw*)]:
fixes f :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: *case-prod* $f \in$ *borel-measurable* $(N \bigotimes_M M)$
shows $(\lambda x. \int y.\ f\ x\ y\ \partial M) \in$ *borel-measurable* N
⟨*proof*⟩

lemma *integrable-product-swap-s-finite*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *M1*:*s-finite-measure M1* **and** *M2*:*s-finite-measure M2*
and *integrable* (*M1* $\bigotimes_M$ *M2*) f
shows *integrable* (*M2* $\bigotimes_M$ *M1*) (λ(*x*,*y*). f (*y*,*x*))
⟨*proof*⟩

lemma *integrable-product-swap-iff-s-finite*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *M1*:*s-finite-measure M1* **and** *M2*:*s-finite-measure M2*
shows *integrable* (*M2* $\bigotimes_M$ *M1*) (λ(*x*,*y*). f (*y*,*x*)) ⟷ *integrable* (*M1* $\bigotimes_M$ *M2*) f
⟨*proof*⟩

lemma *integral-product-swap-s-finite*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *M1*:*s-finite-measure M1* **and** *M2*:*s-finite-measure M2*
and f: f ∈ *borel-measurable* (*M1* $\bigotimes_M$ *M2*)
shows (∫ (*x*,*y*). f (*y*,*x*) ∂(*M2* $\bigotimes_M$ *M1*)) = *integral*^{L} (*M1* $\bigotimes_M$ *M2*) f
⟨*proof*⟩

theorem(**in** *s-finite-measure*) *Fubini-integrable′*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: f ∈ *borel-measurable* (*M1* $\bigotimes_M$ *M*)
and *integ1*: *integrable M1* (λ*x*. ∫ *y*. *norm* (f (*x*, *y*)) ∂*M*)
and *integ2*: *AE x in M1*. *integrable M* (λ*y*. f (*x*, *y*))
shows *integrable* (*M1* $\bigotimes_M$ *M*) f
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *emeasure-pair-measure-finite′*:
assumes *A*: *A* ∈ *sets* (*M1* $\bigotimes_M$ *M*) **and** *finite*: *emeasure* (*M1* $\bigotimes_M$ *M*) *A* < ∞
shows *AE x in M1*. *emeasure M* {*y*∈*space M*. (*x*, *y*) ∈ *A*} < ∞
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *AE-integrable-fst′′′*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M*) f
shows *AE x in M1*. *integrable M* (λ*y*. f (*x*, *y*))
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *integrable-fst-norm′*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M*) f
shows *integrable M1* (λ*x*. ∫ *y*. *norm* (f (*x*, *y*)) ∂*M*)
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *integrable-fst′′′*:
fixes f :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M*) f

shows *integrable M1* ($\lambda x.$ $\int y.$ *f* (*x*, *y*) ∂*M*)
⟨*proof*⟩

proposition(**in** *s-finite-measure*) *integral-fst'''*:
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f*: *integrable* (*M1* $\bigotimes_M$ *M*) *f*
shows ($\int x.$ ($\int y.$ *f* (*x*, *y*) ∂*M*) ∂*M1*) = *integral*^{L} (*M1* $\bigotimes_M$ *M*) *f*
⟨*proof*⟩

lemma (**in** *s-finite-measure*)
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f*: *integrable* (*M1* $\bigotimes_M$ *M*) (*case-prod f*)
shows *AE-integrable-fst''*: *AE x in M1*. *integrable M* ($\lambda y.$ *f x y*)
and *integrable-fst''*: *integrable M1* ($\lambda x.$ $\int y.$ *f x y* ∂*M*)
and *integrable-fst-norm*: *integrable M1* ($\lambda x.$ $\int y.$ *norm* (*f x y*) ∂*M*)
and *integral-fst''*: ($\int x.$ ($\int y.$ *f x y* ∂*M*) ∂*M1*) = *integral*^{L} (*M1* $\bigotimes_M$ *M*) (λ(*x*, *y*). *f x y*)
⟨*proof*⟩

lemma
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes *M1*:*s-finite-measure M1* **and** *M2*:*s-finite-measure M2*
and *f*[*measurable*]: *integrable* (*M1* $\bigotimes_M$ *M2*) (*case-prod f*)
shows *AE-integrable-snd-s-finite*: *AE y in M2*. *integrable M1* ($\lambda x.$ *f x y*) (**is** *?AE*)
and *integrable-snd-s-finite*: *integrable M2* ($\lambda y.$ $\int x.$ *f x y* ∂*M1*) (**is** *?INT*)
and *integrable-snd-norm-s-finite*: *integrable M2* ($\lambda y.$ $\int x.$ *norm* (*f x y*) ∂*M1*) (**is** *?INT2*)
and *integral-snd-s-finite*: ($\int y.$ ($\int x.$ *f x y* ∂*M1*) ∂*M2*) = *integral*^{L} (*M1* $\bigotimes_M$ *M2*) (*case-prod f*) (**is** *?EQ*)
⟨*proof*⟩

proposition *Fubini-integral'*:
fixes *f* :: - ⇒ - ⇒ - :: {*banach*, *second-countable-topology*}
assumes *M1*:*s-finite-measure M1* **and** *M2*:*s-finite-measure M2*
and *f*: *integrable* (*M1* $\bigotimes_M$ *M2*) (*case-prod f*)
shows ($\int y.$ ($\int x.$ *f x y* ∂*M1*) ∂*M2*) = ($\int x.$ ($\int y.$ *f x y* ∂*M2*) ∂*M1*)
⟨*proof*⟩

locale *product-s-finite* =
fixes *M* :: *'i* ⇒ *'a measure*
assumes *s-finite-measures*: $\bigwedge i.$ *s-finite-measure* (*M i*)

sublocale *product-s-finite* ⊆ *M?*: *s-finite-measure M i* **for** *i*
⟨*proof*⟩

locale *finite-product-s-finite* = *product-s-finite M* **for** *M* :: *'i* ⇒ *'a measure* +
fixes *I* :: *'i set*
assumes *finite-index*: *finite I*

lemma (**in** *product-s-finite*) *emeasure-PiM*:
finite I $\Longrightarrow$ ($\bigwedge$*i. i*$\in$*I* $\Longrightarrow$ *A i* $\in$ *sets* (*M i*)) $\Longrightarrow$ *emeasure* (*PiM I M*) (*Pi*$_E$ *I A*) = ($\prod$ *i*$\in$*I. emeasure* (*M i*) (*A i*))
⟨*proof*⟩

lemma (**in** *finite-product-s-finite*) *measure-times*:
($\bigwedge$*i. i* $\in$ *I* $\Longrightarrow$ *A i* $\in$ *sets* (*M i*)) $\Longrightarrow$ *emeasure* (*Pi*$_M$ *I M*) (*Pi*$_E$ *I A*) = ($\prod$ *i*$\in$*I. emeasure* (*M i*) (*A i*))
⟨*proof*⟩

lemma (**in** *product-s-finite*) *nn-integral-empty*:
0 $\leq$ *f* (λ*k. undefined*) $\Longrightarrow$ *integral*^{N} (*Pi*$_M$ {} *M*) *f* = *f* (λ*k. undefined*)
⟨*proof*⟩

Every s-finite measure is represented as the push-forward measure of a σ-finite measure.

definition *Mi-to-NM* :: (*nat* $\Rightarrow$ *'a measure*) $\Rightarrow$ *'a measure* $\Rightarrow$ (*nat* $\times$ *'a*) *measure*
where
Mi-to-NM Mi M $\equiv$ *measure-of* (*space* (*count-space UNIV* $\bigotimes_M$ *M*)) (*sets* (*count-space UNIV* $\bigotimes_M$ *M*)) (λ*A.* $\sum$ *i. distr* (*Mi i*) (*count-space UNIV* $\bigotimes_M$ *M*) (λ*x.* (*i*,*x*)) *A*)

lemma
shows *sets-Mi-to-NM*[*measurable-cong*,*simp*]: *sets* (*Mi-to-NM Mi M*) = *sets* (*count-space UNIV* $\bigotimes_M$ *M*)
and *space-Mi-to-NM*[*simp*]: *space* (*Mi-to-NM Mi M*) = *space* (*count-space UNIV* $\bigotimes_M$ *M*)
⟨*proof*⟩

context
fixes *M* :: *'a measure* **and** *Mi* :: *nat* $\Rightarrow$ *'a measure*
assumes *sets-Mi*[*measurable-cong*,*simp*]: $\bigwedge$*i. sets* (*Mi i*) = *sets M*
and *emeasure-Mi*: $\bigwedge$*A. A* $\in$ *sets M* $\Longrightarrow$ *M A* = ($\sum$ *i. Mi i A*)
begin

lemma *emeasure-Mi-to-NM*:
assumes [*measurable*]: *A* $\in$ *sets* (*count-space UNIV* $\bigotimes_M$ *M*)
shows *emeasure* (*Mi-to-NM Mi M*) *A* = ($\sum$ *i. distr* (*Mi i*) (*count-space UNIV* $\bigotimes_M$ *M*) (λ*x.* (*i*,*x*)) *A*)
⟨*proof*⟩

lemma *sigma-finite-Mi-to-NM-measure*:
assumes $\bigwedge$*i. finite-measure* (*Mi i*)
shows *sigma-finite-measure* (*Mi-to-NM Mi M*)
⟨*proof*⟩

lemma *distr-Mi-to-NM-M*: *distr* (*Mi-to-NM Mi M*) *M snd* = *M*

⟨*proof*⟩

end

context
  fixes μ :: $'a$ *measure*
  assumes *standard-borel-ne*: *standard-borel-ne* μ
    and *s-finite*: *s-finite-measure* μ
begin

interpretation μ : *s-finite-measure* μ ⟨*proof*⟩

interpretation *n-*μ: *standard-borel-ne count-space* (*UNIV* :: *nat set*) $\bigotimes_M$ μ
  ⟨*proof*⟩

lemma *exists-push-forward*:
$\exists (\mu' :: real\ measure)\ f.\ f \in borel \rightarrow_M \mu \wedge sets\ \mu' = sets\ borel \wedge sigma\text{-}finite\text{-}measure\ \mu'$
$\wedge\ distr\ \mu'\ \mu\ f = \mu$
⟨*proof*⟩

abbreviation μ'*-and-f* $\equiv$ (*SOME* (μ'::*real measure*,*f*). $f \in borel \rightarrow_M \mu \wedge sets\ \mu' = sets\ borel \wedge sigma\text{-}finite\text{-}measure\ \mu' \wedge distr\ \mu'\ \mu\ f = \mu$)

definition *sigma-pair-*μ $\equiv$ *fst* μ'*-and-f*
definition *sigma-pair-f* $\equiv$ *snd* μ'*-and-f*

lemma
  shows *sigma-pair-f-measurable* : *sigma-pair-f* $\in borel \rightarrow_M \mu$ (**is** *?g1*)
    and *sets-sigma-pair-*μ: *sets sigma-pair-*μ = *sets borel* (**is** *?g2*)
    and *sigma-finite-sigma-pair-*μ: *sigma-finite-measure sigma-pair-*μ (**is** *?g3*)
    and *distr-sigma-pair*: *distr sigma-pair-*μ μ *sigma-pair-f* = μ (**is** *?g4*)
⟨*proof*⟩

end

definition *s-finite-measure-algebra* :: $'a$ *measure* $\Rightarrow$ $'a$ *measure measure* **where**
  s-finite-measure-algebra K =
    (*SUP A* $\in$ *sets K*. *vimage-algebra* $\{M.$ *s-finite-measure M* $\wedge$ *sets M* = *sets K*$\}$ ($\lambda M.$ *emeasure M A*) *borel*)

lemma *space-s-finite-measure-algebra*:
 space (*s-finite-measure-algebra K*) = $\{M.$ *s-finite-measure M* $\wedge$ *sets M* = *sets K*$\}$
  ⟨*proof*⟩

lemma *s-finite-measure-algebra-cong*: *sets M* = *sets N* $\Longrightarrow$ *s-finite-measure-algebra M* = *s-finite-measure-algebra N*
  ⟨*proof*⟩

lemma *measurable-emeasure-s-finite-measure-algebra*[*measurable*]:
  $a \in sets\ A \Longrightarrow (\lambda M.\ emeasure\ M\ a) \in borel\text{-}measurable\ (s\text{-}finite\text{-}measure\text{-}algebra\ A)$
  ⟨*proof*⟩

lemma *measurable-measure-s-finite-measure-algebra*[*measurable*]:
  $a \in sets\ A \Longrightarrow (\lambda M.\ measure\ M\ a) \in borel\text{-}measurable\ (s\text{-}finite\text{-}measure\text{-}algebra\ A)$
  ⟨*proof*⟩

lemma *s-finite-measure-algebra-measurableD*:
  assumes N: $N \in measurable\ M\ (s\text{-}finite\text{-}measure\text{-}algebra\ S)$ **and** x: $x \in space\ M$
  shows $space\ (N\ x) = space\ S$
    and $sets\ (N\ x) = sets\ S$
    and $measurable\ (N\ x)\ K = measurable\ S\ K$
    and $measurable\ K\ (N\ x) = measurable\ K\ S$
  ⟨*proof*⟩

context
  fixes $K\ M\ N$ **assumes** K: $K \in measurable\ M\ (s\text{-}finite\text{-}measure\text{-}algebra\ N)$
begin

lemma *s-finite-measure-algebra-kernel*: $a \in space\ M \Longrightarrow s\text{-}finite\text{-}measure\ (K\ a)$
  ⟨*proof*⟩

lemma *s-finite-measure-algebra-sets-kernel*: $a \in space\ M \Longrightarrow sets\ (K\ a) = sets\ N$
  ⟨*proof*⟩

lemma *measurable-emeasure-kernel-s-finite-measure-algebra*[*measurable*]:
    $A \in sets\ N \Longrightarrow (\lambda a.\ emeasure\ (K\ a)\ A) \in borel\text{-}measurable\ M$
  ⟨*proof*⟩

end

lemma *measurable-s-finite-measure-algebra*:
  $(\bigwedge a.\ a \in space\ M \Longrightarrow s\text{-}finite\text{-}measure\ (K\ a)) \Longrightarrow$
  $(\bigwedge a.\ a \in space\ M \Longrightarrow sets\ (K\ a) = sets\ N) \Longrightarrow$
  $(\bigwedge A.\ A \in sets\ N \Longrightarrow (\lambda a.\ emeasure\ (K\ a)\ A) \in borel\text{-}measurable\ M) \Longrightarrow$
  $K \in measurable\ M\ (s\text{-}finite\text{-}measure\text{-}algebra\ N)$
  ⟨*proof*⟩

definition *bind-kernel* :: $'a\ measure \Rightarrow ('a \Rightarrow 'b\ measure) \Rightarrow 'b\ measure$ (**infixl** $\gg\!=_k$ *54*) **where**
$bind\text{-}kernel\ M\ k = (if\ space\ M = \{\}\ then\ count\text{-}space\ \{\}\ else$
    $let\ Y = k\ (SOME\ x.\ x \in space\ M)\ in$
     $measure\text{-}of\ (space\ Y)\ (sets\ Y)\ (\lambda B.\ \int^{+} x.\ (k\ x\ B)\ \partial M))$

lemma *bind-kernel-cong-All*:
  assumes $\bigwedge x.\ x \in space\ M \Longrightarrow f\ x = g\ x$

shows $M \gg=_k f = M \gg=_k g$
⟨*proof*⟩

lemma *sets-bind-kernel*:
assumes $\bigwedge x.\ x \in space\ M \Longrightarrow sets\ (k\ x) = sets\ N\ space\ M \neq \{\}$
shows $sets\ (M \gg=_k k) = sets\ N$
⟨*proof*⟩

2.2 Measure Kernel

locale *measure-kernel* =
fixes $X :: 'a\ measure$ **and** $Y :: 'b\ measure$ **and** $\kappa :: 'a \Rightarrow 'b\ measure$
assumes *kernel-sets*[*measurable-cong*]: $\bigwedge x.\ x \in space\ X \Longrightarrow sets\ (\kappa\ x) = sets\ Y$
and *emeasure-measurable*[*measurable*]: $\bigwedge B.\ B \in sets\ Y \Longrightarrow (\lambda x.\ emeasure\ (\kappa\ x)\ B) \in borel\text{-}measurable\ X$
and *Y-not-empty*: $space\ X \neq \{\} \Longrightarrow space\ Y \neq \{\}$
begin

lemma *kernel-space* :$\bigwedge x.\ x \in space\ X \Longrightarrow space\ (\kappa\ x) = space\ Y$
⟨*proof*⟩

lemma *measure-measurable*:
assumes $B \in sets\ Y$
shows $(\lambda x.\ measure\ (\kappa\ x)\ B) \in borel\text{-}measurable\ X$
⟨*proof*⟩

lemma *set-nn-integral-measure*:
assumes [*measurable-cong*]: $sets\ \mu = sets\ X$ **and** [*measurable*]: $A \in sets\ X\ B \in sets\ Y$
defines $\nu \equiv measure\text{-}of\ (space\ Y)\ (sets\ Y)\ (\lambda B.\ \int^{+} x{\in}A.\ (\kappa\ x\ B)\ \partial\mu)$
shows $\nu\ B = (\int^{+} x{\in}A.\ (\kappa\ x\ B)\ \partial\mu)$
⟨*proof*⟩

corollary *nn-integral-measure*:
assumes $sets\ \mu = sets\ X\ B \in sets\ Y$
defines $\nu \equiv measure\text{-}of\ (space\ Y)\ (sets\ Y)\ (\lambda B.\ \int^{+} x.\ (\kappa\ x\ B)\ \partial\mu)$
shows $\nu\ B = (\int^{+} x.\ (\kappa\ x\ B)\ \partial\mu)$
⟨*proof*⟩

lemma *distr-measure-kernel*:
assumes [*measurable*]:$f \in Y \rightarrow_M Z$
shows $measure\text{-}kernel\ X\ Z\ (\lambda x.\ distr\ (\kappa\ x)\ Z\ f)$
⟨*proof*⟩

lemma *measure-kernel-comp*:
assumes [*measurable*]: $f \in W \rightarrow_M X$
shows $measure\text{-}kernel\ W\ Y\ (\lambda x.\ \kappa\ (f\ x))$
⟨*proof*⟩

lemma *emeasure-bind-kernel*:
assumes *sets* μ = *sets* X $B \in$ *sets* Y
shows $(\mu \gg\!=_k \kappa)\ B = (\int^+ x.\ (\kappa\ x\ B)\ \partial\mu)$
⟨*proof*⟩

lemma *measure-bind-kernel*:
assumes [*measurable-cong*]:*sets* μ = *sets* X **and** [*measurable*]:$B \in$ *sets* Y
and *AE* x *in* μ. $\kappa\ x\ B < \infty$
shows *measure* $(\mu \gg\!=_k \kappa)\ B = (\int x.$ *measure* $(\kappa\ x)\ B\ \partial\mu)$
⟨*proof*⟩

lemma *sets-bind-kernel*:
assumes *space* $X \neq \{\}$ *sets* μ = *sets* X
shows *sets* $(\mu \gg\!=_k \kappa)$ = *sets* Y
⟨*proof*⟩

lemma *distr-bind-kernel*:
assumes *space* $X \neq \{\}$ **and** [*measurable-cong*]:*sets* μ = *sets* X **and** [*measurable*]:
$f \in Y \to_M Z$
shows *distr* $(\mu \gg\!=_k \kappa)\ Z\ f = \mu \gg\!=_k (\lambda x.$ *distr* $(\kappa\ x)\ Z\ f)$
⟨*proof*⟩

lemma *bind-kernel-distr*:
assumes [*measurable*]: $f \in W \to_M X$ **and** *space* $W \neq \{\}$
shows *distr* $W\ X\ f \gg\!=_k \kappa = W \gg\!=_k (\lambda x.\ \kappa\ (f\ x))$
⟨*proof*⟩

lemma *bind-kernel-return*:
assumes $x \in$ *space* X
shows *return* $X\ x \gg\!=_k \kappa = \kappa\ x$
⟨*proof*⟩

lemma *nn-integral-measurable-kernel*:
assumes $f \in$ *borel-measurable* Y
shows $(\lambda x.\ (\int^+ y.\ f\ y\ \partial(\kappa\ x))) \in$ *borel-measurable* X
⟨*proof*⟩

corollary *integrable-measurable-kernel*:
fixes f :: $'b \Rightarrow 'c$::{*banach*, *second-countable-topology*}
assumes [*measurable*]:$f \in$ *borel-measurable* Y
shows *Measurable.pred* X $(\lambda x.$ *integrable* $(\kappa\ x)\ f)$
⟨*proof*⟩

lemma *integral-measurable-kernel*:
fixes f :: $'b \Rightarrow 'c$::{*banach*, *second-countable-topology*}
assumes f[*measurable*]: $f \in$ *borel-measurable* Y
shows $(\lambda x.\ (\int y.\ f\ y\ \partial(\kappa\ x))) \in$ *borel-measurable* X
⟨*proof*⟩

lemma *density-measure-kernel′*:
  assumes $f[measurable]$: $f \in Y \rightarrow_M borel$
  shows *measure-kernel* X Y $(\lambda x.\ density\ (\kappa\ x)\ f)$
⟨*proof*⟩

lemma *nn-integral-bind-kernel*:
  assumes $f \in borel\text{-}measurable\ Y$ $sets\ \mu = sets\ X$
  shows $(\int^+ y.\ f\ y\ \partial(\mu \gg\!=_k \kappa)) = (\int^+ x.\ (\int^+ y.\ f\ y\ \partial(\kappa\ x))\ \partial\mu)$
⟨*proof*⟩

lemma *bind-kernel-measure-kernel*:
  assumes *measure-kernel* Y Z k'
  shows *measure-kernel* X Z $(\lambda x.\ \kappa\ x \gg\!=_k k')$
⟨*proof*⟩

lemma *restrict-measure-kernel*: *measure-kernel* (*restrict-space* X A) Y κ
⟨*proof*⟩

end

lemma *measure-kernel-cong-sets*:
  assumes $sets\ X = sets\ X'$ $sets\ Y = sets\ Y'$
  shows *measure-kernel* X Y = *measure-kernel* X' Y'
  ⟨*proof*⟩

lemma *measure-kernel-cong*:
  assumes $\bigwedge x.\ x \in space\ X \Longrightarrow k\ x = k'\ x$
  shows *measure-kernel* X Y k = *measure-kernel* X Y k'
  ⟨*proof*⟩

lemma *measure-kernel-pair-countble1*:
  assumes *countable* A $\bigwedge i.\ i \in A \Longrightarrow$ *measure-kernel* X Y $(\lambda x.\ k\ (i,x))$
  shows *measure-kernel* (*count-space* $A \bigotimes_M X$) Y k
  ⟨*proof*⟩

lemma *measure-kernel-empty-trivial*:
  assumes $space\ X = \{\}$
  shows *measure-kernel* X Y k
  ⟨*proof*⟩

lemma *measure-kernel-const′*: $space\ Y \neq \{\} \Longrightarrow sets\ \mu = sets\ Y \Longrightarrow$ *measure-kernel* X Y $(\lambda r.\ \mu)$
  ⟨*proof*⟩

2.3 Finite Kernel

locale *finite-kernel* = *measure-kernel* +
  assumes *finite-measure-spaces*: $\exists\, r{<}\infty.\ \forall\, x{\in}\ space\ X.\ \kappa\ x\ (space\ Y) < r$

begin

lemma *finite-measures*:
  assumes $x \in$ *space* X
  shows *finite-measure* $(\kappa\ x)$
⟨*proof*⟩

end

lemma *finite-kernel-empty-trivial*: *space* $X = \{\} \Longrightarrow$ *finite-kernel* $X\ Y\ f$
  ⟨*proof*⟩

lemma *finite-kernel-cong-sets*:
  assumes *sets* $X =$ *sets* X' *sets* $Y =$ *sets* Y'
  shows *finite-kernel* $X\ Y =$ *finite-kernel* $X'\ Y'$
  ⟨*proof*⟩

2.4 Sub-Probability Kernel

locale *subprob-kernel* = *measure-kernel* +
  assumes *subprob-spaces*: $\bigwedge x.\ x \in$ *space* $X \Longrightarrow$ *subprob-space* $(\kappa\ x)$
begin
lemma *subprob-space*:
$\bigwedge x.\ x \in$ *space* $X \Longrightarrow \kappa\ x$ (*space* Y) $\leq$ *1*
  ⟨*proof*⟩

lemma *subprob-measurable*[*measurable*]:
 $\kappa \in X \rightarrow_M$ *subprob-algebra* Y
  ⟨*proof*⟩

lemma *finite-kernel*: *finite-kernel* $X\ Y\ \kappa$
  ⟨*proof*⟩

sublocale *finite-kernel*
  ⟨*proof*⟩

end

lemma *subprob-kernel-def'*:
 subprob-kernel $X\ Y\ \kappa \longleftrightarrow \kappa \in X \rightarrow_M$ *subprob-algebra* Y
  ⟨*proof*⟩

lemmas *subprob-kernelI* = *measurable-subprob-algebra*[*simplified subprob-kernel-def'*[*symmetric*]]

lemma *subprob-kernel-cong-sets*:
  assumes *sets* $X =$ *sets* X' *sets* $Y =$ *sets* Y'
  shows *subprob-kernel* $X\ Y =$ *subprob-kernel* $X'\ Y'$
  ⟨*proof*⟩

lemma *subprob-kernel-empty-trivial*:
assumes $space\ X = \{\}$
shows *subprob-kernel X Y k*
⟨*proof*⟩

lemma *bind-kernel-bind*:
assumes $f \in M \rightarrow_M subprob\text{-}algebra\ N$
shows $M \gg=_k f = M \gg= f$
⟨*proof*⟩

lemma(**in** *measure-kernel*) *subprob-kernel-sum*:
assumes $\bigwedge x.\ x \in space\ X \Longrightarrow finite\text{-}measure\ (\kappa\ x)$
obtains *ki* **where** $\bigwedge i.\ subprob\text{-}kernel\ X\ Y\ (ki\ i)\ \bigwedge A\ x.\ x \in space\ X \Longrightarrow \kappa\ x\ A = (\sum i.\ ki\ i\ x\ A)$
⟨*proof*⟩

2.5 Probability Kernel

locale *prob-kernel* = *measure-kernel* +
assumes *prob-spaces*: $\bigwedge x.\ x \in space\ X \Longrightarrow prob\text{-}space\ (\kappa\ x)$
begin

lemma *prob-space*:
$\bigwedge x.\ x \in space\ X \Longrightarrow \kappa\ x\ (space\ Y) = 1$
⟨*proof*⟩

lemma *prob-measurable*[*measurable*]:
$\kappa \in X \rightarrow_M prob\text{-}algebra\ Y$
⟨*proof*⟩

lemma *subprob-kernel*: *subprob-kernel X Y* κ
⟨*proof*⟩

sublocale *subprob-kernel*
⟨*proof*⟩

lemma *restrict-probability-kernel*:
prob-kernel (*restrict-space X A*) *Y* κ
⟨*proof*⟩

end

lemma *prob-kernel-def'*:
$prob\text{-}kernel\ X\ Y\ \kappa \longleftrightarrow \kappa \in X \rightarrow_M prob\text{-}algebra\ Y$
⟨*proof*⟩

lemma *bind-kernel-return''*:
assumes $sets\ M = sets\ N$

shows $M \gg=_k$ *return* $N = M$
$\langle proof \rangle$

2.6 S-Finite Kernel

locale *s-finite-kernel* = *measure-kernel* +
assumes *s-finite-kernel-sum*: $\exists ki. (\forall i.$ *finite-kernel* $X\ Y\ (ki\ i) \wedge (\forall x \in space\ X.\ \forall A \in sets\ Y.\ \kappa\ x\ A = (\sum i.\ ki\ i\ x\ A)))$

lemma *s-finite-kernel-subI*:
assumes $\bigwedge x.\ x \in space\ X \Longrightarrow sets\ (\kappa\ x) = sets\ Y$ $\bigwedge i.$ *subprob-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x\ A.\ x \in space\ X \Longrightarrow A \in sets\ Y \Longrightarrow emeasure\ (\kappa\ x)\ A = (\sum i.\ ki\ i\ x\ A)$
shows *s-finite-kernel* $X\ Y\ \kappa$
$\langle proof \rangle$

context *s-finite-kernel*
begin

lemma *s-finite-kernels-fin*:
obtains *ki* **where** $\bigwedge i.$ *finite-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x\ A.\ x \in space\ X \Longrightarrow \kappa\ x\ A = (\sum i.\ ki\ i\ x\ A)$
$\langle proof \rangle$

lemma *s-finite-kernels*:
obtains *ki* **where** $\bigwedge i.$ *subprob-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x\ A.\ x \in space\ X \Longrightarrow \kappa\ x\ A = (\sum i.\ ki\ i\ x\ A)$
$\langle proof \rangle$

lemma *image-s-finite-measure*:
assumes $x \in space\ X$
shows *s-finite-measure* $(\kappa\ x)$
$\langle proof \rangle$

corollary *kernel-measurable-s-finite*[*measurable*]:$\kappa \in X \rightarrow_M$ *s-finite-measure-algebra* Y
$\langle proof \rangle$

lemma *comp-measurable*:
assumes f[*measurable*]:$f \in M \rightarrow_M X$
shows *s-finite-kernel* $M\ Y\ (\lambda x.\ \kappa\ (f\ x))$
$\langle proof \rangle$

lemma *distr-s-finite-kernel*:
assumes f[*measurable*]: $f \in Y \rightarrow_M Z$
shows *s-finite-kernel* $X\ Z\ (\lambda x.\ distr\ (\kappa\ x)\ Z\ f)$
$\langle proof \rangle$

lemma *comp-s-finite-measure*:
assumes *s-finite-measure* μ **and** [*measurable-cong*]: $sets\ \mu = sets\ X$

shows *s-finite-measure* $(\mu \ggg_k \kappa)$
⟨*proof*⟩

end

lemma *s-finite-kernel-empty-trivial*:
assumes *space* $X = \{\}$
shows *s-finite-kernel* $X\ Y\ k$
⟨*proof*⟩

lemma *s-finite-kernel-def'*: *s-finite-kernel* $X\ Y\ \kappa \longleftrightarrow ((\forall x.\ x \in$ *space* $X \longrightarrow$ *sets* $(\kappa\ x) =$ *sets* $Y) \wedge (\exists\, ki.\ (\forall\, i.$ *subprob-kernel* $X\ Y\ (ki\ i)) \wedge (\forall\, x\ A.\ x \in$ *space* $X \longrightarrow A \in$ *sets* $Y \longrightarrow$ *emeasure* $(\kappa\ x)\ A = (\sum i.\ ki\ i\ x\ A))))$ (**is** *?l* $\longleftrightarrow$ *?r*)
⟨*proof*⟩

lemma(**in** *finite-kernel*) *s-finite-kernel-finite-kernel*: *s-finite-kernel* $X\ Y\ \kappa$
⟨*proof*⟩

lemmas(**in** *subprob-kernel*) *s-finite-kernel-subprob-kernel* = *s-finite-kernel-finite-kernel*
lemmas(**in** *prob-kernel*) *s-finite-kernel-prob-kernel* = *s-finite-kernel-subprob-kernel*

sublocale *finite-kernel* ⊆ *s-finite-kernel*
⟨*proof*⟩

lemma *s-finite-kernel-cong-sets*:
assumes *sets* $X =$ *sets* X' *sets* $Y =$ *sets* Y'
shows *s-finite-kernel* $X\ Y =$ *s-finite-kernel* $X'\ Y'$
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *s-finite-kernel-cong*:
assumes $\bigwedge x.\ x \in$ *space* $X \Longrightarrow \kappa\ x = g\ x$
shows *s-finite-kernel* $X\ Y\ g$
⟨*proof*⟩

lemma(**in** *s-finite-measure*) *s-finite-kernel-const*:
assumes *space* $M \neq \{\}$
shows *s-finite-kernel* $X\ M\ (\lambda x.\ M)$
⟨*proof*⟩

corollary(**in** *s-finite-measure*) *s-finite-kernel-const'*:
assumes *sets* $M =$ *sets* N *space* $N \neq \{\}$
shows *s-finite-kernel* $X\ N\ (\lambda x.\ M)$
⟨*proof*⟩

lemma *s-finite-kernel-pair-countble1*:
assumes *countable* A $\bigwedge i.\ i \in A \Longrightarrow$ *s-finite-kernel* $X\ Y\ (\lambda x.\ k\ (i,x))$
shows *s-finite-kernel* (*count-space* $A \bigotimes_M X)\ Y\ k$
⟨*proof*⟩

lemma *s-finite-kernel-s-finite-kernel*:
assumes $\bigwedge i.$ *s-finite-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x.\ x \in$ *space* $X \Longrightarrow$ *sets* $(k\ x) =$ *sets* Y
$\bigwedge x\ A.\ x \in$ *space* $X \Longrightarrow A \in$ *sets* $Y \Longrightarrow$ *emeasure* $(k\ x)\ A = (\sum i.\ (ki\ i)\ x\ A)$
shows *s-finite-kernel* $X\ Y\ k$
⟨*proof*⟩

lemma *s-finite-kernel-finite-sumI*:
assumes [*measurable-cong*]: $\bigwedge x.\ x \in$ *space* $X \Longrightarrow$ *sets* $(\kappa\ x) =$ *sets* Y
and $\bigwedge i.\ i \in I \Longrightarrow$ *subprob-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x\ A.\ x \in$ *space* $X \Longrightarrow A \in$ *sets* $Y \Longrightarrow$ *emeasure* $(\kappa\ x)\ A = (\sum i{\in}I.\ ki\ i\ x\ A)$ *finite* $I\ I \neq \{\}$
shows *s-finite-kernel* $X\ Y\ \kappa$
⟨*proof*⟩

Each kernel does not need to be bounded by a uniform upper-bound in the definition of *s-finite-kernel*

lemma *s-finite-kernel-finite-bounded-sum*:
assumes [*measurable-cong*]: $\bigwedge x.\ x \in$ *space* $X \Longrightarrow$ *sets* $(\kappa\ x) =$ *sets* Y
and $\bigwedge i.$ *measure-kernel* $X\ Y\ (ki\ i)$ $\bigwedge x\ A.\ x \in$ *space* $X \Longrightarrow A \in$ *sets* $Y \Longrightarrow$ $\kappa\ x\ A = (\sum i.\ ki\ i\ x\ A)$ $\bigwedge i\ x.\ x \in$ *space* $X \Longrightarrow ki\ i\ x\ ($*space* $Y) < \infty$
shows *s-finite-kernel* $X\ Y\ \kappa$
⟨*proof*⟩

lemma(**in** *measure-kernel*) *s-finite-kernel-finite-bounded*:
assumes $\bigwedge x.\ x \in$ *space* $X \Longrightarrow \kappa\ x\ ($*space* $Y) < \infty$
shows *s-finite-kernel* $X\ Y\ \kappa$
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *density-s-finite-kernel*:
assumes f[*measurable*]: *case-prod* $f \in X \bigotimes_M Y \to_M$ *borel*
shows *s-finite-kernel* $X\ Y\ (\lambda x.$ *density* $(\kappa\ x)\ (f\ x))$
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *nn-integral-measurable-f*:
assumes [*measurable*]:$(\lambda(x,y).\ f\ x\ y) \in$ *borel-measurable* $(X \bigotimes_M Y)$
shows $(\lambda x.\ \int^+ y.\ f\ x\ y\ \partial(\kappa\ x)) \in$ *borel-measurable* X
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *nn-integral-measurable-f'*:
assumes $f \in$ *borel-measurable* $(X \bigotimes_M Y)$
shows $(\lambda x.\ \int^+ y.\ f\ (x,\ y)\ \partial(\kappa\ x)) \in$ *borel-measurable* X
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *bind-kernel-s-finite-kernel'*:
assumes *s-finite-kernel* $(X \bigotimes_M Y)\ Z$ (*case-prod* g)
shows *s-finite-kernel* $X\ Z\ (\lambda x.\ \kappa\ x \gg\!=_k g\ x)$
⟨*proof*⟩

corollary(**in** *s-finite-kernel*) *bind-kernel-s-finite-kernel*:
assumes *s-finite-kernel* $Y\ Z\ k'$

shows *s-finite-kernel X Z* ($\lambda x.\ \kappa\ x \gg=_k k'$)
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *bind-kernel-assoc*:
assumes *s-finite-kernel Y Z k' sets* μ = *sets X*
shows $\mu \gg=_k (\lambda x.\ \kappa\ x \gg=_k k') = \mu \gg=_k \kappa \gg=_k k'$
⟨*proof*⟩

lemma *s-finite-kernel-pair-measure*:
assumes *s-finite-kernel X Y k* **and** *s-finite-kernel X Z k'*
shows *s-finite-kernel X* ($Y \bigotimes_M Z$) ($\lambda x.\ k\ x \bigotimes_M k'\ x$)
⟨*proof*⟩

lemma *pair-measure-eq-bind-s-finite*:
assumes *s-finite-measure* μ *s-finite-measure* ν
shows $\mu \bigotimes_M \nu = \mu \gg=_k (\lambda x.\ \nu \gg=_k (\lambda y.\ return\ (\mu \bigotimes_M \nu)\ (x,y)))$
⟨*proof*⟩

lemma *bind-kernel-rotate-return*:
assumes *s-finite-measure* μ *s-finite-measure* ν
shows $\mu \gg=_k (\lambda x.\ \nu \gg=_k (\lambda y.\ return\ (\mu \bigotimes_M \nu)\ (x,y))) = \nu \gg=_k (\lambda y.\ \mu \gg=_k (\lambda x.\ return\ (\mu \bigotimes_M \nu)\ (x,y)))$
⟨*proof*⟩

lemma *bind-kernel-rotate'*:
assumes *s-finite-measure* μ *s-finite-measure* ν *s-finite-kernel* ($\mu \bigotimes_M \nu$) *Z* (*case-prod f*)
shows $\mu \gg=_k (\lambda x.\ \nu \gg=_k (\lambda y.\ f\ x\ y)) = \nu \gg=_k (\lambda y.\ \mu \gg=_k (\lambda x.\ f\ x\ y))$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *bind-kernel-rotate*:
assumes *sets* μ = *sets X* **and** *sets* ν = *sets Y*
and *s-finite-measure* μ *s-finite-measure* ν *s-finite-kernel* ($X \bigotimes_M Y$) *Z* ($\lambda(x,y).\ f\ x\ y$)
shows $\mu \gg=_k (\lambda x.\ \nu \gg=_k (\lambda y.\ f\ x\ y)) = \nu \gg=_k (\lambda y.\ \mu \gg=_k (\lambda x.\ f\ x\ y))$
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *emeasure-measurable'*:
assumes *A*[*measurable*]: (*SIGMA x:space X. A x*) ∈ *sets* ($X \bigotimes_M Y$)
shows ($\lambda x.$ *emeasure* (κ *x*) (*A x*)) ∈ *borel-measurable X*
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *measure-measurable'*:
assumes (*SIGMA x:space X. A x*) ∈ *sets* ($X \bigotimes_M Y$)
shows ($\lambda x.$ *measure* (κ *x*) (*A x*)) ∈ *borel-measurable X*
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *AE-pred*:

assumes *P*[*measurable*]:*Measurable.pred* ($X \bigotimes_M Y$) (*case-prod P*)
shows *Measurable.pred X* (λx. *AE y in* κ *x*. *P x y*)
⟨*proof*⟩

lemma(**in** *subprob-kernel*) *integrable-probability-kernel-pred*:
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:($\lambda(x,y)$. *f x y*) ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows *Measurable.pred X* (λx. *integrable* (κ *x*) (*f x*))
⟨*proof*⟩

corollary *integrable-measurable-subprob′*:
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:($\lambda(x,y)$. *f x y*) ∈ *borel-measurable* ($X \bigotimes_M Y$) *k* ∈ $X \to_M$ *subprob-algebra Y*
shows *Measurable.pred X* (λx. *integrable* (*k x*) (*f x*))
⟨*proof*⟩

lemma(**in** *subprob-kernel*) *integrable-probability-kernel-pred′*:
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes *f* ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows *Measurable.pred X* (λx. *integrable* (κ *x*) (*curry f x*))
⟨*proof*⟩

lemma(**in** *subprob-kernel*) *lebesgue-integral-measurable-f-subprob*:
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:*f* ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows (λx. $\int$ *y*. *f* (*x*,*y*) ∂(κ *x*)) ∈ *borel-measurable X*
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *integrable-measurable-pred*[*measurable* (*raw*)]:
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:*case-prod f* ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows *Measurable.pred X* (λx. *integrable* (κ *x*) (*f x*))
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *integral-measurable-f*:
fixes *f* :: - ⇒ - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:*case-prod f* ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows (λx. $\int$ *y*. *f x y* ∂(κ *x*)) ∈ *borel-measurable X*
⟨*proof*⟩

lemma(**in** *s-finite-kernel*) *integral-measurable-f′*:
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable*]:*f* ∈ *borel-measurable* ($X \bigotimes_M Y$)
shows (λx. $\int$ *y*. *f* (*x*,*y*) ∂(κ *x*)) ∈ *borel-measurable X*
⟨*proof*⟩

lemma(**in** *measure-kernel*)
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}

assumes [*measurable-cong*]: *sets* μ = *sets X*
and *integrable* ($\mu \ggg_k \kappa$) *f*
shows *integrable-bind-kernelD1*: *integrable* μ ($\lambda x.\ \int y.$ *norm* (*f y*) $\partial\kappa$ *x*) (**is** *?g1*)
and *integrable-bind-kernelD1'*: *integrable* μ ($\lambda x.\ \int y.$ *f y* $\partial\kappa$ *x*) (**is** *?g1'*)
and *integrable-bind-kernelD2*: *AE x in* μ. *integrable* (κ *x*) *f* (**is** *?g2*)
and *integrable-bind-kernelD3*: *space X* $\neq$ {} $\Longrightarrow$ *f* $\in$ *borel-measurable Y* (**is** - $\Longrightarrow$ *?g3*)
⟨*proof*⟩

lemma(**in** *measure-kernel*)
fixes *f* :: - ⇒ -::{*banach*, *second-countable-topology*}
assumes [*measurable-cong*]: *sets* μ = *sets X*
and [*measurable*]:*AE x in* μ. *integrable* (κ *x*) *f integrable* μ ($\lambda x.\ \int y.$ *norm* (*f y*) $\partial\kappa$ *x*) *f* $\in$ *borel-measurable Y*
shows *integrable-bind-kernel*: *integrable* ($\mu \ggg_k \kappa$) *f*
and *integral-bind-kernel*: ($\int y.$ *f y* $\partial(\mu \ggg_k \kappa)$) = ($\int x.$ ($\int y.$ *f y*$\partial\kappa$ *x*)∂ μ) (**is** *?eq*)
⟨*proof*⟩

end

3 Quasi-Borel Spaces

theory *QuasiBorel*
imports *HOL−Probability.Probability*
begin

3.1 Definitions

3.1.1 Quasi-Borel Spaces

definition *qbs-closed1* :: (*real* ⇒ *'a*) *set* ⇒ *bool*
where *qbs-closed1 Mx* ≡ (∀ *a* ∈ *Mx*. ∀ *f* ∈ (*borel* :: *real measure*) $\rightarrow_M$ (*borel* :: *real measure*). *a* ∘ *f* ∈ *Mx*)

definition *qbs-closed2* :: [*'a set*, (*real* ⇒ *'a*) *set*] ⇒ *bool*
where *qbs-closed2 X Mx* ≡ (∀ *x* ∈ *X*. (λ*r*. *x*) ∈ *Mx*)

definition *qbs-closed3* :: (*real* ⇒ *'a*) *set* ⇒ *bool*
where *qbs-closed3 Mx* ≡ (∀ *P*::*real* ⇒ *nat*. ∀ *Fi*::*nat* ⇒ *real* ⇒ *'a*.
(*P* ∈ *borel* $\rightarrow_M$ *count-space UNIV*) ⟶ (∀ *i*. *Fi i* ∈ *Mx*) ⟶ (λ*r*. *Fi* (*P r*) *r*) ∈ *Mx*)

lemma *separate-measurable*:
fixes *P* :: *real* ⇒ *nat*
assumes ⋀*i*. *P* −‘ {*i*} ∈ *sets borel*
shows *P* ∈ *borel* $\rightarrow_M$ *count-space UNIV*
⟨*proof*⟩

lemma *measurable-separate*:
  fixes P :: *real* $\Rightarrow$ *nat*
  assumes $P \in$ *borel* $\to_M$ *count-space UNIV*
  shows $P -\text{`}\ \{i\} \in$ *sets borel*
  ⟨*proof*⟩

definition *is-quasi-borel X Mx* $\longleftrightarrow$ *Mx* $\subseteq$ *UNIV* $\to$ *X* $\wedge$ *qbs-closed1 Mx* $\wedge$ *qbs-closed2 X Mx* $\wedge$ *qbs-closed3 Mx*

lemma *is-quasi-borel-intro*[*simp*]:
  assumes *Mx* $\subseteq$ *UNIV* $\to$ *X*
      and *qbs-closed1 Mx qbs-closed2 X Mx qbs-closed3 Mx*
    shows *is-quasi-borel X Mx*
  ⟨*proof*⟩

typedef *'a quasi-borel* = {(*X*::*'a set*, *Mx*). *is-quasi-borel X Mx*}
⟨*proof*⟩

definition *qbs-space* :: *'a quasi-borel* $\Rightarrow$ *'a set* **where**
  qbs-space X $\equiv$ *fst* (*Rep-quasi-borel X*)

definition *qbs-Mx* :: *'a quasi-borel* $\Rightarrow$ (*real* $\Rightarrow$ *'a*) *set* **where**
  qbs-Mx X $\equiv$ *snd* (*Rep-quasi-borel X*)

declare [[*coercion qbs-space*]]

lemma *qbs-decomp* : (*qbs-space X*,*qbs-Mx X*) $\in$ {(*X*::*'a set*, *Mx*). *is-quasi-borel X Mx*}
  ⟨*proof*⟩

lemma *qbs-Mx-to-X*:
  assumes $\alpha \in$ *qbs-Mx X*
  shows α *r* $\in$ *qbs-space X*
  ⟨*proof*⟩

lemma *qbs-closed1I*:
  assumes $\bigwedge\alpha\ f.\ \alpha \in Mx \Longrightarrow f \in$ *borel* $\to_M$ *borel* $\Longrightarrow \alpha \circ f \in Mx$
  shows *qbs-closed1 Mx*
  ⟨*proof*⟩

lemma *qbs-closed1-dest*[*simp*]:
  assumes $\alpha \in$ *qbs-Mx X*
      and $f \in$ *borel* $\to_M$ *borel*
    shows $\alpha \circ f \in$ *qbs-Mx X*
  ⟨*proof*⟩

lemma *qbs-closed1-dest'*[*simp*]:
  assumes $\alpha \in$ *qbs-Mx X*

and $f \in$ *borel* $\rightarrow_M$ *borel*
shows $(\lambda r.\ \alpha\ (f\ r)) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed2I*:
assumes $\bigwedge x.\ x \in X \Longrightarrow (\lambda r.\ x) \in Mx$
shows *qbs-closed2 X Mx*
⟨*proof*⟩

lemma *qbs-closed2-dest*[*simp*]:
assumes $x \in$ *qbs-space X*
shows $(\lambda r.\ x) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3I*:
assumes $\bigwedge(P :: real \Rightarrow nat)\ Fi.\ P \in borel \rightarrow_M count\text{-}space\ UNIV \Longrightarrow (\bigwedge i.\ Fi\ i \in Mx)$
$\Longrightarrow (\lambda r.\ Fi\ (P\ r)\ r) \in Mx$
shows *qbs-closed3 Mx*
⟨*proof*⟩

lemma *qbs-closed3I'*:
assumes $\bigwedge(P :: real \Rightarrow nat)\ Fi.\ (\bigwedge i.\ P -` \{i\} \in sets\ borel) \Longrightarrow (\bigwedge i.\ Fi\ i \in Mx)$
$\Longrightarrow (\lambda r.\ Fi\ (P\ r)\ r) \in Mx$
shows *qbs-closed3 Mx*
⟨*proof*⟩

lemma *qbs-closed3-dest*[*simp*]:
fixes $P{::}real \Rightarrow nat$ **and** $Fi :: nat \Rightarrow real \Rightarrow$ -
assumes $P \in borel \rightarrow_M$ *count-space UNIV*
and $\bigwedge i.\ Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest'*:
fixes $P{::}real \Rightarrow nat$ **and** $Fi :: nat \Rightarrow real \Rightarrow$ -
assumes $\bigwedge i.\ P -` \{i\} \in sets\ borel$
and $\bigwedge i.\ Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest2*:
assumes *countable I*
and [*measurable*]: $P \in borel \rightarrow_M$ *count-space I*
and $\bigwedge i.\ i \in I \Longrightarrow Fi\ i \in$ *qbs-Mx X*
shows $(\lambda r.\ Fi\ (P\ r)\ r) \in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-closed3-dest2'*:
assumes *countable I*
and [*measurable*]: $P \in$ *borel* $\rightarrow_M$ *count-space I*
and $\bigwedge i.\ i \in$ *range P* $\Longrightarrow$ *Fi i* $\in$ *qbs-Mx X*
shows ($\lambda r.$ *Fi* (*P r*) *r*) $\in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-Mx-indicat*:
assumes $S \in$ *sets borel* $\alpha \in$ *qbs-Mx X* $\beta \in$ *qbs-Mx X*
shows ($\lambda r.$ *if* $r \in S$ *then* α *r else* β *r*) $\in$ *qbs-Mx X*
⟨*proof*⟩

lemma *qbs-space-Mx*: *qbs-space X* = {α *x* |*x* α. $\alpha \in$ *qbs-Mx X*}
⟨*proof*⟩

lemma *qbs-space-eq-Mx*:
assumes *qbs-Mx X* = *qbs-Mx Y*
shows *qbs-space X* = *qbs-space Y*
⟨*proof*⟩

lemma *qbs-eqI*:
assumes *qbs-Mx X* = *qbs-Mx Y*
shows *X* = *Y*
⟨*proof*⟩

3.1.2 Empty Space

definition *empty-quasi-borel* :: *'a quasi-borel* **where**
empty-quasi-borel ≡ *Abs-quasi-borel* ({},{})

lemma
shows *eqb-space*[*simp*]: *qbs-space empty-quasi-borel* = ({} :: *'a set*)
and *eqb-Mx*[*simp*]: *qbs-Mx empty-quasi-borel* = ({} :: (*real* ⇒ *'a*) *set*)
⟨*proof*⟩

lemma *qbs-empty-equiv* :*qbs-space X* = {} ⟷ *qbs-Mx X* = {}
⟨*proof*⟩

lemma *empty-quasi-borel-iff*:
qbs-space X = {} ⟷ *X* = *empty-quasi-borel*
⟨*proof*⟩

3.1.3 Unit Space

definition *unit-quasi-borel* :: *unit quasi-borel* (1_Q) **where**
unit-quasi-borel ≡ *Abs-quasi-borel* (*UNIV*,*UNIV*)

lemma
shows *unit-qbs-space*[*simp*]: *qbs-space unit-quasi-borel* = {()}
and *unit-qbs-Mx*[*simp*]: *qbs-Mx unit-quasi-borel* = {$\lambda r.$ ()}

⟨*proof*⟩

3.1.4 Sub-Spaces

definition *sub-qbs* :: $['a\ \mathit{quasi\text{-}borel}, 'a\ \mathit{set}] \Rightarrow 'a\ \mathit{quasi\text{-}borel}$ **where**
$\mathit{sub\text{-}qbs}\ X\ U \equiv \mathit{Abs\text{-}quasi\text{-}borel}\ (\mathit{qbs\text{-}space}\ X \cap U,\{\alpha.\ \alpha \in \mathit{qbs\text{-}Mx}\ X \wedge (\forall r.\ \alpha\ r \in U)\})$

lemma
shows *sub-qbs-space*: $\mathit{qbs\text{-}space}\ (\mathit{sub\text{-}qbs}\ X\ U) = \mathit{qbs\text{-}space}\ X \cap U$
and *sub-qbs-Mx*: $\mathit{qbs\text{-}Mx}\ (\mathit{sub\text{-}qbs}\ X\ U) = \{\alpha.\ \alpha \in \mathit{qbs\text{-}Mx}\ X \wedge (\forall r.\ \alpha\ r \in U)\}$
⟨*proof*⟩

lemma *sub-qbs*:
assumes $U \subseteq \mathit{qbs\text{-}space}\ X$
shows $(\mathit{qbs\text{-}space}\ (\mathit{sub\text{-}qbs}\ X\ U),\ \mathit{qbs\text{-}Mx}\ (\mathit{sub\text{-}qbs}\ X\ U)) = (U,\ \{f \in \mathit{UNIV} \to U.\ f \in \mathit{qbs\text{-}Mx}\ X\})$
⟨*proof*⟩

lemma *sub-qbs-ident*: $\mathit{sub\text{-}qbs}\ X\ (\mathit{qbs\text{-}space}\ X) = X$
⟨*proof*⟩

lemma *sub-qbs-sub-qbs*: $\mathit{sub\text{-}qbs}\ (\mathit{sub\text{-}qbs}\ X\ A)\ B = \mathit{sub\text{-}qbs}\ X\ (A \cap B)$
⟨*proof*⟩

3.1.5 Image Spaces

definition *map-qbs* :: $['a \Rightarrow 'b] \Rightarrow 'a\ \mathit{quasi\text{-}borel} \Rightarrow 'b\ \mathit{quasi\text{-}borel}$ **where**
$\mathit{map\text{-}qbs}\ f\ X = \mathit{Abs\text{-}quasi\text{-}borel}\ (f\ `\ (\mathit{qbs\text{-}space}\ X),\{f \circ \alpha\ |\alpha.\ \alpha \in \mathit{qbs\text{-}Mx}\ X\})$

lemma
shows *map-qbs-space*: $\mathit{qbs\text{-}space}\ (\mathit{map\text{-}qbs}\ f\ X) = f\ `\ (\mathit{qbs\text{-}space}\ X)$
and *map-qbs-Mx*: $\mathit{qbs\text{-}Mx}\ (\mathit{map\text{-}qbs}\ f\ X) = \{f \circ \alpha\ |\alpha.\ \alpha \in \mathit{qbs\text{-}Mx}\ X\}$
⟨*proof*⟩

3.1.6 Binary Product Spaces

definition *pair-qbs* :: $['a\ \mathit{quasi\text{-}borel}, 'b\ \mathit{quasi\text{-}borel}] \Rightarrow ('a \times 'b)\ \mathit{quasi\text{-}borel}$ (**infixr** $\bigotimes_Q$ *80*) **where**
$\mathit{pair\text{-}qbs}\ X\ Y = \mathit{Abs\text{-}quasi\text{-}borel}\ (\mathit{qbs\text{-}space}\ X \times \mathit{qbs\text{-}space}\ Y,\ \{f.\ \mathit{fst} \circ f \in \mathit{qbs\text{-}Mx}\ X \wedge \mathit{snd} \circ f \in \mathit{qbs\text{-}Mx}\ Y\})$

lemma
shows *pair-qbs-space*: $\mathit{qbs\text{-}space}\ (X \bigotimes_Q Y) = \mathit{qbs\text{-}space}\ X \times \mathit{qbs\text{-}space}\ Y$
and *pair-qbs-Mx*: $\mathit{qbs\text{-}Mx}\ (X \bigotimes_Q Y) = \{f.\ \mathit{fst} \circ f \in \mathit{qbs\text{-}Mx}\ X \wedge \mathit{snd} \circ f \in \mathit{qbs\text{-}Mx}\ Y\}$
⟨*proof*⟩

lemma *pair-qbs-fst*:
assumes $\mathit{qbs\text{-}space}\ Y \neq \{\}$

shows *map-qbs fst* $(X \bigotimes_Q Y) = X$
⟨*proof*⟩

lemma *pair-qbs-snd*:
assumes *qbs-space* $X \neq \{\}$
shows *map-qbs snd* $(X \bigotimes_Q Y) = Y$
⟨*proof*⟩

3.1.7 Binary Coproduct Spaces

definition *copair-qbs-Mx* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*real* => *'a* + *'b*) *set*
where
copair-qbs-Mx X Y ≡
{*g*. ∃ *S* ∈ *sets borel*.
(*S* = {} ⟶ (∃ *α1*∈ *qbs-Mx X*. *g* = (λ*r*. *Inl* (*α1 r*)))) ∧
(*S* = *UNIV* ⟶ (∃ *α2*∈ *qbs-Mx Y*. *g* = (λ*r*. *Inr* (*α2 r*)))) ∧
((*S* ≠ {} ∧ *S* ≠ *UNIV*) ⟶
(∃ *α1*∈ *qbs-Mx X*. ∃ *α2*∈ *qbs-Mx Y*.
g = (λ*r*::*real*. (*if* (*r* ∈ *S*) *then Inl* (*α1 r*) *else Inr* (*α2 r*)))))}

definition *copair-qbs* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*'a* + *'b*) *quasi-borel*
(**infixr** $\bigoplus_Q$ *65*) **where**
copair-qbs X Y ≡ *Abs-quasi-borel* (*qbs-space X* <+> *qbs-space Y*, *copair-qbs-Mx X Y*)

The following is an equivalent definition of *copair-qbs-Mx*.

definition *copair-qbs-Mx2* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*real* => *'a* + *'b*) *set* **where**
copair-qbs-Mx2 X Y ≡
{*g*. (*if qbs-space X* = {} ∧ *qbs-space Y* = {} *then False*
else if qbs-space X ≠ {} ∧ *qbs-space Y* = {} *then*
(∃*α1*∈ *qbs-Mx X*. *g* = (λ*r*. *Inl* (*α1 r*)))
else if qbs-space X = {} ∧ *qbs-space Y* ≠ {} *then*
(∃*α2*∈ *qbs-Mx Y*. *g* = (λ*r*. *Inr* (*α2 r*)))
else
(∃*S* ∈ *sets borel*. ∃*α1*∈ *qbs-Mx X*. ∃*α2*∈ *qbs-Mx Y*.
g = (λ*r*::*real*. (*if* (*r* ∈ *S*) *then Inl* (*α1 r*) *else Inr* (*α2 r*))))) }

lemma *copair-qbs-Mx-equiv* :*copair-qbs-Mx* (*X* :: *'a quasi-borel*) (*Y* :: *'b quasi-borel*) = *copair-qbs-Mx2 X Y*
⟨*proof*⟩

lemma
shows *copair-qbs-space*: *qbs-space* $(X \bigoplus_Q Y)$ = *qbs-space X* <+> *qbs-space Y* (**is** *?goal1*)
and *copair-qbs-Mx*: *qbs-Mx* $(X \bigoplus_Q Y)$ = *copair-qbs-Mx X Y* (**is** *?goal2*)
⟨*proof*⟩

lemma *copair-qbs-MxD*:

assumes $g \in$ *qbs-Mx* $(X \bigoplus_Q Y)$
and $\bigwedge\alpha.\ \alpha \in$ *qbs-Mx* $X \Longrightarrow g = (\lambda r.$ *Inl* $(\alpha\ r)) \Longrightarrow P\ g$
and $\bigwedge\beta.\ \beta \in$ *qbs-Mx* $Y \Longrightarrow g = (\lambda r.$ *Inr* $(\beta\ r)) \Longrightarrow P\ g$
and $\bigwedge S\ \alpha\ \beta.\ (S ::$ *real set*$) \in$ *sets borel* $\Longrightarrow S \neq \{\} \Longrightarrow S \neq$ *UNIV* $\Longrightarrow \alpha \in$ *qbs-Mx* $X \Longrightarrow \beta \in$ *qbs-Mx* $Y \Longrightarrow g = (\lambda r.$ *if* $r \in S$ *then Inl* $(\alpha\ r)$ *else Inr* $(\beta\ r)) \Longrightarrow P\ g$
shows $P\ g$
⟨*proof*⟩

3.1.8 Product Spaces

definition *PiQ* :: *'a set* ⇒ (*'a* ⇒ *'b quasi-borel*) ⇒ (*'a* ⇒ *'b*) *quasi-borel* **where**
PiQ I X ≡ *Abs-quasi-borel* $(\Pi_E\ i{\in}I.$ *qbs-space* $(X\ i), \{\alpha.\ \forall i.\ (i \in I \longrightarrow (\lambda r.\ \alpha\ r\ i) \in$ *qbs-Mx* $(X\ i)) \wedge (i \notin I \longrightarrow (\lambda r.\ \alpha\ r\ i) = (\lambda r.$ *undefined*$))\})$

syntax
-PiQ :: *pttrn* ⇒ *'i set* ⇒ *'a quasi-borel* ⇒ (*'i* => *'a*) *quasi-borel* ((*3*Π_Q -∈-./ -) *10*)
syntax-consts
-PiQ == *PiQ*
translations
Π_Q *x*∈*I*. *X* == *CONST PiQ I* (λ*x*. *X*)

lemma
shows *PiQ-space*: *qbs-space* (*PiQ I X*) = $(\Pi_E\ i{\in}I.$ *qbs-space* $(X\ i))$ (**is** *?goal1*)
and *PiQ-Mx*: *qbs-Mx* (*PiQ I X*) = $\{\alpha.\ \forall i.\ (i \in I \longrightarrow (\lambda r.\ \alpha\ r\ i) \in$ *qbs-Mx* $(X\ i)) \wedge (i \notin I \longrightarrow (\lambda r.\ \alpha\ r\ i) = (\lambda r.$ *undefined*$))\}$ (**is** - = *?Mx*)
⟨*proof*⟩

lemma *prod-qbs-MxI*:
assumes $\bigwedge i.\ i \in I \Longrightarrow (\lambda r.\ \alpha\ r\ i) \in$ *qbs-Mx* $(X\ i)$
and $\bigwedge i.\ i \notin I \Longrightarrow (\lambda r.\ \alpha\ r\ i) = (\lambda r.$ *undefined*$)$
shows $\alpha \in$ *qbs-Mx* (*PiQ I X*)
⟨*proof*⟩

lemma *prod-qbs-MxD*:
assumes $\alpha \in$ *qbs-Mx* (*PiQ I X*)
shows $\bigwedge i.\ i \in I \Longrightarrow (\lambda r.\ \alpha\ r\ i) \in$ *qbs-Mx* $(X\ i)$
and $\bigwedge i.\ i \notin I \Longrightarrow (\lambda r.\ \alpha\ r\ i) = (\lambda r.$ *undefined*$)$
and $\bigwedge i\ r.\ i \notin I \Longrightarrow \alpha\ r\ i =$ *undefined*
⟨*proof*⟩

lemma *PiQ-eqI*:
assumes $\bigwedge i.\ i \in I \Longrightarrow X\ i = Y\ i$
shows *PiQ I X* = *PiQ I Y*
⟨*proof*⟩

lemma *PiQ-empty*: *qbs-space* (*PiQ* {} *X*) = {λ*i*. *undefined*}
⟨*proof*⟩

lemma *PiQ-empty-Mx*: *qbs-Mx* (*PiQ* {} *X*) = {λr *i*. *undefined*}
⟨*proof*⟩

3.1.9 Coproduct Spaces

definition *coPiQ-Mx* :: [*'a set*, *'a* ⇒ *'b quasi-borel*] ⇒ (*real* ⇒ *'a* × *'b*) *set* **where**
coPiQ-Mx I X ≡ { λr. (*f r*, α (*f r*) *r*) |*f* α. *f* ∈ *borel* $\rightarrow_M$ *count-space I* ∧ (∀ *i*∈*range f*. α *i* ∈ *qbs-Mx* (*X i*))}

definition *coPiQ-Mx'* :: [*'a set*, *'a* ⇒ *'b quasi-borel*] ⇒ (*real* ⇒ *'a* × *'b*) *set* **where**
coPiQ-Mx' I X ≡ { λr. (*f r*, α (*f r*) *r*) |*f* α. *f* ∈ *borel* $\rightarrow_M$ *count-space I* ∧ (∀ *i*. (*i* ∈ *range f* ∨ *qbs-space* (*X i*) ≠ {}) ⟶ α *i* ∈ *qbs-Mx* (*X i*))}

lemma *coPiQ-Mx-eq*:
coPiQ-Mx I X = *coPiQ-Mx' I X*
⟨*proof*⟩

definition *coPiQ* :: [*'a set*, *'a* ⇒ *'b quasi-borel*] ⇒ (*'a* × *'b*) *quasi-borel* **where**
coPiQ I X ≡ *Abs-quasi-borel* (*SIGMA i*:*I*. *qbs-space* (*X i*), *coPiQ-Mx I X*)

syntax
-coPiQ :: *pttrn* ⇒ *'i set* ⇒ *'a quasi-borel* ⇒ (*'i* × *'a*) *quasi-borel* ((*3*$\amalg_Q$ -∈-./ -) *10*)
syntax-consts
-coPiQ ⇌ *coPiQ*
translations
$\amalg_Q$ *x*∈*I*. *X* ⇌ *CONST coPiQ I* (λx. *X*)

lemma
shows *coPiQ-space*: *qbs-space* (*coPiQ I X*) = (*SIGMA i*:*I*. *qbs-space* (*X i*)) (**is** *?goal1*)
and *coPiQ-Mx*: *qbs-Mx* (*coPiQ I X*) = *coPiQ-Mx I X* (**is** *?goal2*)
⟨*proof*⟩

lemma *coPiQ-MxI*:
assumes *f* ∈ *borel* $\rightarrow_M$ *count-space I*
and ⋀*i*. *i* ∈ *range f* ⟹ α *i* ∈ *qbs-Mx* (*X i*)
shows (λr. (*f r*, α (*f r*) *r*)) ∈ *qbs-Mx* (*coPiQ I X*)
⟨*proof*⟩

lemma *coPiQ-eqI*:
assumes ⋀*i*. *i* ∈ *I* ⟹ *X i* = *Y i*
shows *coPiQ I X* = *coPiQ I Y*
⟨*proof*⟩

3.1.10 List Spaces

We define the quasi-Borel spaces on list using the following isomorphism.

$$List(X) \cong \coprod_{n\in\mathbb{N}} \prod_{0\leq i<n} X$$

definition *list-nil* :: *nat* × (*nat* ⇒ *'a*) **where**
list-nil ≡ (*0*, *λn. undefined*)
definition *list-cons* :: [*'a*, *nat* × (*nat* ⇒ *'a*)] ⇒ *nat* × (*nat* ⇒ *'a*) **where**
list-cons x l ≡ (*Suc* (*fst l*), (*λn. if n* = *0 then x else* (*snd l*) (*n* − *1*)))

fun *from-list* :: *'a list* ⇒ *nat* × (*nat* ⇒ *'a*) **where**
from-list [] = *list-nil* |
from-list (*a*#*l*) = *list-cons a* (*from-list l*)

fun *to-list'* :: *nat* ⇒ (*nat* ⇒ *'a*) ⇒ *'a list* **where**
to-list' 0 - = [] |
to-list' (*Suc n*) *f* = *f 0* # *to-list' n* (*λn. f* (*Suc n*))

definition *to-list* :: *nat* × (*nat* ⇒ *'a*) ⇒ *'a list* **where**
to-list ≡ *case-prod to-list'*

lemma *inj-on-to-list*: *inj-on* (*to-list* :: *nat* × (*nat* ⇒ *'a*) ⇒ *'a list*) (*SIGMA n*:*UNIV*. *PiE* {..<*n*} *A*)
⟨*proof*⟩

Definition

definition *list-qbs* :: *'a quasi-borel* ⇒ *'a list quasi-borel* **where**
list-qbs X ≡ *map-qbs to-list* (Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *X*)

definition *list-head* :: *nat* × (*nat* ⇒ *'a*) ⇒ *'a* **where**
list-head l = *snd l 0*
definition *list-tail* :: *nat* × (*nat* ⇒ *'a*) ⇒ *nat* × (*nat* ⇒ *'a*) **where**
list-tail l = (*fst l* − *1*, *λm.* (*snd l*) (*Suc m*))

lemma *list-simp1*: *list-nil* ≠ *list-cons x l*
⟨*proof*⟩

lemma *list-simp2*:
assumes *list-cons a al* = *list-cons b bl*
shows *a* = *b al* = *bl*
⟨*proof*⟩

lemma
shows *list-simp3*:*list-head* (*list-cons a l*) = *a*
and *list-simp4*:*list-tail* (*list-cons a l*) = *l*
⟨*proof*⟩

lemma *list-decomp1*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
shows l = *list-nil* $\vee$
($\exists\, a\ l'$. $a \in$ *qbs-space* X $\wedge$ $l' \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X) $\wedge$ l = *list-cons* a l')
⟨*proof*⟩

lemma *list-simp5*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
and $l \neq$ *list-nil*
shows l = *list-cons* (*list-head* l) (*list-tail* l)
⟨*proof*⟩

lemma *list-simp6*:
list-nil $\in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
⟨*proof*⟩

lemma *list-simp7*:
assumes $a \in$ *qbs-space* X
and $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
shows *list-cons* a $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
⟨*proof*⟩

lemma *list-destruct-rule*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
P *list-nil*
and $\bigwedge a\ l'$. $a \in$ *qbs-space* $X \Longrightarrow l' \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X) $\Longrightarrow P$ (*list-cons* a l')
shows P l
⟨*proof*⟩

lemma *list-induct-rule*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
P *list-nil*
and $\bigwedge a\ l'$. $a \in$ *qbs-space* $X \Longrightarrow l' \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X) $\Longrightarrow P\ l' \Longrightarrow P$ (*list-cons* a l')
shows P l
⟨*proof*⟩

lemma *to-list-simp1*: *to-list list-nil* = []
⟨*proof*⟩

lemma *to-list-simp2*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)
shows *to-list* (*list-cons* a l) = a # *to-list* l
⟨*proof*⟩

lemma *to-list-set*:
assumes $l \in$ *qbs-space* ($\amalg_Q$ $n\in$(*UNIV* :: *nat set*).Π_Q $i\in\{..<n\}$. X)

shows *set* (*to-list* *l*) ⊆ *qbs-space* *X*
⟨*proof*⟩

lemma *from-list-length*: *fst* (*from-list* *l*) = *length* *l*
⟨*proof*⟩

lemma *from-list-in-list-of*:
assumes *set* *l* ⊆ *qbs-space* *X*
shows *from-list* *l* ∈ *qbs-space* (Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *X*)
⟨*proof*⟩

lemma *from-list-in-list-of'*: *from-list* *l* ∈ *qbs-space* ((Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *Abs-quasi-borel* (*UNIV*,*UNIV*)))
⟨*proof*⟩

lemma *list-cons-in-list-of*:
assumes *set* (*a*#*l*) ⊆ *qbs-space* *X*
shows *list-cons* *a* (*from-list* *l*) ∈ *qbs-space* (Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *X*)
⟨*proof*⟩

lemma *from-list-to-list-ident*:
to-list (*from-list* *l*) = *l*
⟨*proof*⟩

lemma *to-list-from-list-ident*:
assumes *l* ∈ *qbs-space* (Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *X*)
shows *from-list* (*to-list* *l*) = *l*
⟨*proof*⟩

definition *rec-list'* :: *'b* ⇒ (*'a* ⇒ (*nat* × (*nat* ⇒ *'a*)) ⇒ *'b* ⇒ *'b*) ⇒ (*nat* × (*nat* ⇒ *'a*)) ⇒ *'b* **where**
rec-list' *t0* *f* *l* ≡ (*rec-list* *t0* (λ*x* *l'*. *f* *x* (*from-list* *l'*)) (*to-list* *l*))

lemma *rec-list'-simp1*:
rec-list' *t* *f* *list-nil* = *t*
⟨*proof*⟩

lemma *rec-list'-simp2*:
assumes *l* ∈ *qbs-space* (Π_Q *n*∈(*UNIV* :: *nat set*).Π_Q *i*∈{..<*n*}. *X*)
shows *rec-list'* *t* *f* (*list-cons* *x* *l*) = *f* *x* *l* (*rec-list'* *t* *f* *l*)
⟨*proof*⟩

lemma *list-qbs-space*: *qbs-space* (*list-qbs* *X*) = *lists* (*qbs-space* *X*)
⟨*proof*⟩

3.1.11 Option Spaces

The option spaces is defined using the following isomorphism.

$$Option(X) \cong X + 1$$

definition *option-qbs* :: *'a quasi-borel* ⇒ *'a option quasi-borel* **where**
option-qbs X = *map-qbs* (λ*x*. *case x of Inl y* ⇒ *Some y* | *Inr y* ⇒ *None*) ($X \bigoplus_Q$ 1_Q)

lemma *option-qbs-space*: *qbs-space* (*option-qbs X*) = {*Some x*|*x*. *x* ∈ *qbs-space X*} ∪ {*None*}
⟨*proof*⟩

3.1.12 Function Spaces

definition *exp-qbs* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*'a* ⇒ *'b*) *quasi-borel* (**infixr** $\Rightarrow_Q$ *61*) **where**
$X \Rightarrow_Q Y$ ≡ *Abs-quasi-borel* ({*f*. ∀α ∈ *qbs-Mx X*. *f* ∘ α ∈ *qbs-Mx Y*}, {*g*. ∀α∈ *borel-measurable borel*. ∀β∈ *qbs-Mx X*. (λ*r*. *g* (α *r*) (β *r*)) ∈ *qbs-Mx Y*})

lemma
shows *exp-qbs-space*: *qbs-space* (*exp-qbs X Y*) = {*f*. ∀α ∈ *qbs-Mx X*. *f* ∘ α ∈ *qbs-Mx Y*}
and *exp-qbs-Mx*: *qbs-Mx* (*exp-qbs X Y*) = {*g*. ∀α∈ *borel-measurable borel*. ∀β∈ *qbs-Mx X*. (λ*r*. *g* (α *r*) (β *r*)) ∈ *qbs-Mx Y*}
⟨*proof*⟩

3.1.13 Ordering on Quasi-Borel Spaces

inductive-set *generating-Mx* :: *'a set* ⇒ (*real* ⇒ *'a*) *set* ⇒ (*real* ⇒ *'a*) *set*
for *X* :: *'a set* **and** *Mx* :: (*real* ⇒ *'a*) *set*
where
Basic: α ∈ *Mx* ⟹ α ∈ *generating-Mx X Mx*
| *Const*: *x* ∈ *X* ⟹ (λ*r*. *x*) ∈ *generating-Mx X Mx*
| *Comp* : *f* ∈ (*borel* :: *real measure*) $\rightarrow_M$ (*borel* :: *real measure*) ⟹ α ∈ *generating-Mx X Mx* ⟹ α ∘ *f* ∈ *generating-Mx X Mx*
| *Part* : (⋀*i*. *Fi i* ∈ *generating-Mx X Mx*) ⟹ *P* ∈ *borel* $\rightarrow_M$ *count-space* (*UNIV* :: *nat set*) ⟹ (λ*r*. *Fi* (*P r*) *r*) ∈ *generating-Mx X Mx*

lemma *generating-Mx-to-space*:
assumes *Mx* ⊆ *UNIV* → *X*
shows *generating-Mx X Mx* ⊆ *UNIV* → *X*
⟨*proof*⟩

lemma *generating-Mx-closed1*:
qbs-closed1 (*generating-Mx X Mx*)
⟨*proof*⟩

lemma *generating-Mx-closed2*:

qbs-closed2 X (generating-Mx X Mx)
⟨*proof*⟩

lemma *generating-Mx-closed3*:
qbs-closed3 (generating-Mx X Mx)
⟨*proof*⟩

lemma *generating-Mx-Mx*:
generating-Mx (qbs-space X) (qbs-Mx X) = qbs-Mx X
⟨*proof*⟩

instantiation *quasi-borel* :: (*type*) *order-bot*
begin

inductive *less-eq-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *bool* **where**
qbs-space X ⊂ *qbs-space Y* ⟹ *less-eq-quasi-borel X Y*
| *qbs-space X* = *qbs-space Y* ⟹ *qbs-Mx Y* ⊆ *qbs-Mx X* ⟹ *less-eq-quasi-borel X Y*

lemma *le-quasi-borel-iff*:
X ≤ *Y* ⟷ (*if qbs-space X* = *qbs-space Y then qbs-Mx Y* ⊆ *qbs-Mx X else qbs-space X* ⊂ *qbs-space Y*)
⟨*proof*⟩

definition *less-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *bool* **where**
less-quasi-borel X Y ⟷ (*X* ≤ *Y* ∧ ¬ *Y* ≤ *X*)

definition *bot-quasi-borel* :: *'a quasi-borel* **where**
bot-quasi-borel = *empty-quasi-borel*

instance
⟨*proof*⟩
end

definition *inf-quasi-borel* :: [*'a quasi-borel*, *'a quasi-borel*] ⇒ *'a quasi-borel* **where**
inf-quasi-borel X X' = *Abs-quasi-borel* (*qbs-space X* ∩ *qbs-space X'*, *qbs-Mx X* ∩ *qbs-Mx X'*)

lemma *inf-quasi-borel-correct*: *Rep-quasi-borel* (*inf-quasi-borel X X'*) = (*qbs-space X* ∩ *qbs-space X'*, *qbs-Mx X* ∩ *qbs-Mx X'*)
⟨*proof*⟩

lemma *inf-qbs-space*[*simp*]: *qbs-space* (*inf-quasi-borel X X'*) = *qbs-space X* ∩ *qbs-space X'*
⟨*proof*⟩

lemma *inf-qbs-Mx*[*simp*]: *qbs-Mx* (*inf-quasi-borel X X'*) = *qbs-Mx X* ∩ *qbs-Mx X'*
⟨*proof*⟩

definition *max-quasi-borel* :: *'a set* ⇒ *'a quasi-borel* **where**
max-quasi-borel X = *Abs-quasi-borel* (*X*, *UNIV* → *X*)

lemma *max-quasi-borel-correct*: *Rep-quasi-borel* (*max-quasi-borel X*) = (*X*, *UNIV* → *X*)
⟨*proof*⟩

lemma *max-qbs-space*[*simp*]: *qbs-space* (*max-quasi-borel X*) = *X*
⟨*proof*⟩

lemma *max-qbs-Mx*[*simp*]: *qbs-Mx* (*max-quasi-borel X*) = *UNIV* → *X*
⟨*proof*⟩

instantiation *quasi-borel* :: (*type*) *semilattice-sup*
begin

definition *sup-quasi-borel* :: *'a quasi-borel* ⇒ *'a quasi-borel* ⇒ *'a quasi-borel* **where**
sup-quasi-borel X Y ≡ (*if qbs-space X* = *qbs-space Y* *then inf-quasi-borel X Y*
else if qbs-space X ⊂ *qbs-space Y then Y*
else if qbs-space Y ⊂ *qbs-space X then X*
else max-quasi-borel (*qbs-space X* ∪ *qbs-space Y*))

instance
⟨*proof*⟩

end

end

3.2 Morphisms of Quasi-Borel Spaces

theory *QBS-Morphism*

imports
QuasiBorel

begin

abbreviation *qbs-morphism* :: [*'a quasi-borel*, *'b quasi-borel*] ⇒ (*'a* ⇒ *'b*) *set* (**infixr** $\rightarrow_Q$ *60*) **where**
$X \rightarrow_Q Y$ ≡ *qbs-space* ($X \Rightarrow_Q Y$)

lemma *qbs-morphismI*: (⋀α. α ∈ *qbs-Mx X* ⟹ *f* ∘ α ∈ *qbs-Mx Y*) ⟹ *f* ∈ $X \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-def*: $X \rightarrow_Q Y$ = {*f*∈*qbs-space X* → *qbs-space Y*. ∀α ∈ *qbs-Mx X*. *f* ∘ α ∈ *qbs-Mx Y*}

⟨*proof*⟩

lemma *qbs-morphism-Mx*:
assumes $f \in X \rightarrow_Q Y$ $\alpha \in$ *qbs-Mx* X
shows $f \circ \alpha \in$ *qbs-Mx* Y
⟨*proof*⟩

lemma *qbs-morphism-space*:
assumes $f \in X \rightarrow_Q Y$ $x \in$ *qbs-space* X
shows $f\,x \in$ *qbs-space* Y
⟨*proof*⟩

lemma *qbs-morphism-ident*[*simp*]:
id $\in X \rightarrow_Q X$
⟨*proof*⟩

lemma *qbs-morphism-ident'*[*simp*]:
$(\lambda x.\ x) \in X \rightarrow_Q X$
⟨*proof*⟩

lemma *qbs-morphism-comp*:
assumes $f \in X \rightarrow_Q Y$ $g \in Y \rightarrow_Q Z$
shows $g \circ f \in X \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-compose-rev*:
assumes $f \in Y \rightarrow_Q Z$ **and** $g \in X \rightarrow_Q Y$
shows $(\lambda x.\ f\ (g\ x)) \in X \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-compose*:
assumes $g \in X \rightarrow_Q Y$ **and** $f \in Y \rightarrow_Q Z$
shows $(\lambda x.\ f\ (g\ x)) \in X \rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-cong'*:
assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\,x = g\,x$
and $f \in X \rightarrow_Q Y$
shows $g \in X \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-cong*:
assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\,x = g\,x$
shows $f \in X \rightarrow_Q Y \longleftrightarrow g \in X \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-const*:
assumes $y \in$ *qbs-space* Y
shows $(\lambda x.\ y) \in X \rightarrow_Q Y$

⟨*proof*⟩

lemma *qbs-morphism-from-empty*: *qbs-space* $X = \{\} \Longrightarrow f \in X \to_Q Y$
⟨*proof*⟩

lemma *unit-quasi-borel-terminal*: $\exists!\ f.\ f \in X \to_Q$ *unit-quasi-borel*
⟨*proof*⟩

definition *to-unit-quasi-borel* :: $'a \Rightarrow$ *unit* $(!_Q)$ **where**
to-unit-quasi-borel $\equiv (\lambda r.())$

lemma *to-unit-quasi-borel-morphism*:
$!_Q \in X \to_Q$ *unit-quasi-borel*
⟨*proof*⟩

lemma *qbs-morphism-subD*:
assumes $f \in X \to_Q$ *sub-qbs* $Y\ A$
shows $f \in X \to_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-subI1*:
assumes $f \in X \to_Q Y$ $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x \in A$
shows $f \in X \to_Q$ *sub-qbs* $Y\ A$
⟨*proof*⟩

lemma *qbs-morphism-subI2*:
assumes $f \in X \to_Q Y$
shows $f \in$ *sub-qbs* $X\ A \to_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-subI2'*:
assumes $f \in X \to_Q Y$ *qbs-space* $Z \subseteq$ *qbs-space* X *qbs-Mx* $Z \subseteq$ *qbs-Mx* X
shows $f \in Z \to_Q Y$
⟨*proof*⟩

corollary *qbs-morphism-subsubI*:
assumes $f \in X \to_Q Y$ $\bigwedge x.\ x \in A \Longrightarrow x \in$ *qbs-space* $X \Longrightarrow f\ x \in B$
shows $f \in$ *sub-qbs* $X\ A \to_Q$ *sub-qbs* $Y\ B$
⟨*proof*⟩

lemma *map-qbs-morphism-f*: $f \in X \to_Q$ *map-qbs* $f\ X$
⟨*proof*⟩

lemma *map-qbs-morphism-inverse-f*:
assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow g\ (f\ x) = x$
shows $g \in$ *map-qbs* $f\ X \to_Q X$
⟨*proof*⟩

lemma *pair-qbs-morphismI*:

assumes $\bigwedge\alpha\ \beta.\ \alpha \in \mathit{qbs\text{-}Mx}\ X \Longrightarrow \beta \in \mathit{qbs\text{-}Mx}\ Y$
$\Longrightarrow (\lambda r.\ f\ (\alpha\ r,\ \beta\ r)) \in \mathit{qbs\text{-}Mx}\ Z$
shows $f \in (X \bigotimes_Q\ Y) \to_Q\ Z$
⟨*proof*⟩

lemma *pair-qbs-MxD*:
assumes $\gamma \in \mathit{qbs\text{-}Mx}\ (X \bigotimes_Q\ Y)$
obtains $\alpha\ \beta$ **where** $\alpha \in \mathit{qbs\text{-}Mx}\ X\ \beta \in \mathit{qbs\text{-}Mx}\ Y\ \gamma = (\lambda x.\ (\alpha\ x,\ \beta\ x))$
⟨*proof*⟩

lemma *pair-qbs-MxI*:
assumes $(\lambda x.\ \mathit{fst}\ (\gamma\ x)) \in \mathit{qbs\text{-}Mx}\ X$ **and** $(\lambda x.\ \mathit{snd}\ (\gamma\ x)) \in \mathit{qbs\text{-}Mx}\ Y$
shows $\gamma \in \mathit{qbs\text{-}Mx}\ (X \bigotimes_Q\ Y)$
⟨*proof*⟩

lemma
shows *fst-qbs-morphism*: $\mathit{fst} \in X \bigotimes_Q\ Y \to_Q\ X$
and *snd-qbs-morphism*: $\mathit{snd} \in X \bigotimes_Q\ Y \to_Q\ Y$
⟨*proof*⟩

lemma *qbs-morphism-pair-iff*:
$f \in X \to_Q\ Y \bigotimes_Q\ Z \longleftrightarrow \mathit{fst} \circ f \in X \to_Q\ Y \wedge \mathit{snd} \circ f \in X \to_Q\ Z$
⟨*proof*⟩

lemma *qbs-morphism-Pair*:
assumes $f \in Z \to_Q\ X$
and $g \in Z \to_Q\ Y$
shows $(\lambda z.\ (f\ z,\ g\ z)) \in Z \to_Q\ X \bigotimes_Q\ Y$
⟨*proof*⟩

lemma *qbs-morphism-curry*: $\mathit{curry} \in \mathit{exp\text{-}qbs}\ (X \bigotimes_Q\ Y)\ Z \to_Q \mathit{exp\text{-}qbs}\ X\ (\mathit{exp\text{-}qbs}\ Y\ Z)$
⟨*proof*⟩

corollary *curry-preserves-morphisms*:
assumes $(\lambda xy.\ f\ (\mathit{fst}\ xy)\ (\mathit{snd}\ xy)) \in X \bigotimes_Q\ Y \to_Q\ Z$
shows $f \in X \to_Q\ Y \Rightarrow_Q\ Z$
⟨*proof*⟩

lemma *qbs-morphism-eval*:
$(\lambda fx.\ (\mathit{fst}\ fx)\ (\mathit{snd}\ fx)) \in (X \Rightarrow_Q\ Y) \bigotimes_Q\ X \to_Q\ Y$
⟨*proof*⟩

corollary *qbs-morphism-app*:
assumes $f \in X \to_Q\ (Y \Rightarrow_Q\ Z)\ g \in X \to_Q\ Y$
shows $(\lambda x.\ (f\ x)\ (g\ x)) \in X \to_Q\ Z$
⟨*proof*⟩

⟨*ML*⟩

declare
fst-qbs-morphism[*qbs*]
snd-qbs-morphism[*qbs*]
qbs-morphism-const[*qbs*]
qbs-morphism-ident[*qbs*]
qbs-morphism-ident'[*qbs*]
qbs-morphism-curry[*qbs*]

lemma [*qbs*]:
shows *qbs-morphism-Pair1*: $Pair \in X \to_Q Y \Rightarrow_Q (X \bigotimes_Q Y)$
⟨*proof*⟩

lemma *qbs-morphism-case-prod*[*qbs*]: $case\text{-}prod \in exp\text{-}qbs\ X\ (exp\text{-}qbs\ Y\ Z) \to_Q exp\text{-}qbs\ (X \bigotimes_Q Y)\ Z$
⟨*proof*⟩

lemma *uncurry-preserves-morphisms*:
assumes [*qbs*]:$(\lambda x\ y.\ f\ (x,y)) \in X \to_Q Y \Rightarrow_Q Z$
shows $f \in X \bigotimes_Q Y \to_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-comp'*[*qbs*]:$comp \in Y \Rightarrow_Q Z \to_Q (X \Rightarrow_Q Y) \Rightarrow_Q X \Rightarrow_Q Z$
⟨*proof*⟩

lemma *arg-swap-morphism*:
assumes $f \in X \to_Q exp\text{-}qbs\ Y\ Z$
shows $(\lambda y\ x.\ f\ x\ y) \in Y \to_Q exp\text{-}qbs\ X\ Z$
⟨*proof*⟩

lemma *exp-qbs-comp-morphism*:
assumes $f \in W \to_Q exp\text{-}qbs\ X\ Y$
and $g \in W \to_Q exp\text{-}qbs\ Y\ Z$
shows $(\lambda w.\ g\ w \circ f\ w) \in W \to_Q exp\text{-}qbs\ X\ Z$
⟨*proof*⟩

lemma *arg-swap-morphism-map-qbs1*:
assumes $g \in exp\text{-}qbs\ W\ (exp\text{-}qbs\ X\ Y) \to_Q Z$
shows $(\lambda k.\ g\ (k \circ f)) \in exp\text{-}qbs\ (map\text{-}qbs\ f\ W)\ (exp\text{-}qbs\ X\ Y) \to_Q Z$
⟨*proof*⟩

lemma *qbs-morphism-map-prod*[*qbs*]: $map\text{-}prod \in X \Rightarrow_Q Y \to_Q (W \Rightarrow_Q Z) \Rightarrow_Q (X \bigotimes_Q W) \Rightarrow_Q (Y \bigotimes_Q Z)$
⟨*proof*⟩

lemma *qbs-morphism-pair-swap*:
assumes $f \in X \bigotimes_Q Y \to_Q Z$
shows $(\lambda(x,y).\ f\ (y,x)) \in Y \bigotimes_Q X \to_Q Z$
⟨*proof*⟩

lemma
shows *qbs-morphism-pair-assoc1*: $(\lambda((x,y),z).\ (x,(y,z))) \in (X \bigotimes_Q Y) \bigotimes_Q Z \to_Q X \bigotimes_Q (Y \bigotimes_Q Z)$
and *qbs-morphism-pair-assoc2*: $(\lambda(x,(y,z)).\ ((x,y),z)) \in X \bigotimes_Q (Y \bigotimes_Q Z) \to_Q (X \bigotimes_Q Y) \bigotimes_Q Z$
⟨*proof*⟩

lemma *Inl-qbs-morphism*[*qbs*]: $Inl \in X \to_Q X \bigoplus_Q Y$
⟨*proof*⟩

lemma *Inr-qbs-morphism*[*qbs*]: $Inr \in Y \to_Q X \bigoplus_Q Y$
⟨*proof*⟩

lemma *case-sum-qbs-morphism*[*qbs*]: $\textit{case-sum} \in X \Rightarrow_Q Z \to_Q (Y \Rightarrow_Q Z) \Rightarrow_Q (X \bigoplus_Q Y \Rightarrow_Q Z)$
⟨*proof*⟩

lemma *map-sum-qbs-morphism*[*qbs*]: $\textit{map-sum} \in X \Rightarrow_Q Y \to_Q (X' \Rightarrow_Q Y') \Rightarrow_Q (X \bigoplus_Q X' \Rightarrow_Q Y \bigoplus_Q Y')$
⟨*proof*⟩

lemma *qbs-morphism-component-singleton*[*qbs*]:
assumes $i \in I$
shows $(\lambda x.\ x\ i) \in (\Pi_Q\ i{\in}I.\ (M\ i)) \to_Q M\ i$
⟨*proof*⟩

lemma *qbs-morphism-component-singleton'*:
assumes $f \in Y \to_Q (\Pi_Q\ i{\in}I.\ X\ i)$ $g \in Z \to_Q Y$ $i \in I$
shows $(\lambda x.\ f\ (g\ x)\ i) \in Z \to_Q X\ i$
⟨*proof*⟩

lemma *product-qbs-canonical1*:
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i \in Y \to_Q X\ i$
and $\bigwedge i.\ i \notin I \Longrightarrow f\ i = (\lambda y.\ \textit{undefined})$
shows $(\lambda y\ i.\ f\ i\ y) \in Y \to_Q (\Pi_Q\ i{\in}I.\ X\ i)$
⟨*proof*⟩

lemma *product-qbs-canonical2*:
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i \in Y \to_Q X\ i$
$\bigwedge i.\ i \notin I \Longrightarrow f\ i = (\lambda y.\ \textit{undefined})$
$g \in Y \to_Q (\Pi_Q\ i{\in}I.\ X\ i)$
$\bigwedge i.\ i \in I \Longrightarrow f\ i = (\lambda x.\ x\ i) \circ g$
and $y \in \textit{qbs-space}\ Y$
shows $g\ y = (\lambda i.\ f\ i\ y)$
⟨*proof*⟩

lemma *merge-qbs-morphism*:
$\textit{merge}\ I\ J \in (\Pi_Q\ i{\in}I.\ (M\ i)) \bigotimes_Q (\Pi_Q\ j{\in}J.\ (M\ j)) \to_Q (\Pi_Q\ i{\in}I \cup J.\ (M\ i))$

⟨*proof*⟩

lemma *ini-morphism*[*qbs*]:
  assumes $j \in I$
  shows $(\lambda x.\ (j,x)) \in X\ j \to_Q (\Pi_Q\ i{\in}I.\ X\ i)$
  ⟨*proof*⟩

lemma *coPiQ-canonical1*:
  assumes *countable I*
    and $\bigwedge i.\ i \in I \Longrightarrow f\ i \in X\ i \to_Q Y$
  shows $(\lambda(i,x).\ f\ i\ x) \in (\Pi_Q\ i \in I.\ X\ i) \to_Q Y$
⟨*proof*⟩

lemma *coPiQ-canonical1'*:
  assumes *countable I*
    and $\bigwedge i.\ i \in I \Longrightarrow (\lambda x.\ f\ (i,x)) \in X\ i \to_Q Y$
  shows $f \in (\Pi_Q\ i \in I.\ X\ i) \to_Q Y$
  ⟨*proof*⟩

lemma *None-qbs*[*qbs*]: *None* ∈ *qbs-space* (*option-qbs X*)
  ⟨*proof*⟩

lemma *Some-qbs*[*qbs*]: *Some* ∈ $X \to_Q$ *option-qbs X*
⟨*proof*⟩

lemma *case-option-qbs-morphism*[*qbs*]: *case-option* ∈ *qbs-space* ($Y \Rightarrow_Q (X \Rightarrow_Q Y) \Rightarrow_Q$ *option-qbs X* $\Rightarrow_Q Y$)
⟨*proof*⟩

lemma *rec-option-qbs-morphism*[*qbs*]: *rec-option* ∈ *qbs-space* ($Y \Rightarrow_Q (X \Rightarrow_Q Y) \Rightarrow_Q$ *option-qbs X* $\Rightarrow_Q Y$)
⟨*proof*⟩

lemma *bind-option-qbs-morphism*[*qbs*]: (⤜) ∈ *qbs-space* (*option-qbs X* $\Rightarrow_Q$ ($X \Rightarrow_Q$ *option-qbs Y*) $\Rightarrow_Q$ *option-qbs Y*)
  ⟨*proof*⟩

lemma *Let-qbs-morphism*[*qbs*]: *Let* ∈ $X \Rightarrow_Q (X \Rightarrow_Q Y) \Rightarrow_Q Y$
⟨*proof*⟩

end

3.3 Relation to Measurable Spaces

theory *Measure-QuasiBorel-Adjunction*
  imports *QuasiBorel QBS-Morphism Lemmas-S-Finite-Measure-Monad*
begin

We construct the adjunction between **Meas** and **QBS**, where **Meas** is the category of measurable spaces and measurable functions, and **QBS** is the category of quasi-Borel spaces and morphisms.

3.3.1 The Functor R

definition *measure-to-qbs* :: *'a measure* ⇒ *'a quasi-borel* **where**
measure-to-qbs $X \equiv$ *Abs-quasi-borel* (*space* X, *borel* $\rightarrow_M$ X)

declare [[*coercion measure-to-qbs*]]

lemma
 shows *qbs-space-R*: *qbs-space* (*measure-to-qbs* X) = *space* X (**is** *?goal1*)
  and *qbs-Mx-R*: *qbs-Mx* (*measure-to-qbs* X) = *borel* $\rightarrow_M$ X (**is** *?goal2*)
⟨*proof*⟩

The following lemma says that *measure-to-qbs* is a functor from **Meas** to **QBS**.

lemma *r-preserves-morphisms*:
 $X \rightarrow_M Y \subseteq$ (*measure-to-qbs* X) $\rightarrow_Q$ (*measure-to-qbs* Y)
 ⟨*proof*⟩

lemma *measurable-imp-qbs-morphism*: $f \in M \rightarrow_M N \Longrightarrow f \in M \rightarrow_Q N$
 ⟨*proof*⟩

3.3.2 The Functor L

definition *sigma-Mx* :: *'a quasi-borel* ⇒ *'a set set* **where**
sigma-Mx $X \equiv \{U \cap$ *qbs-space* X $|U. \forall \alpha{\in}$*qbs-Mx* X. α −' $U \in$ *sets borel*$\}$

definition *qbs-to-measure* :: *'a quasi-borel* ⇒ *'a measure* **where**
qbs-to-measure $X \equiv$ *Abs-measure* (*qbs-space* X, *sigma-Mx* X, λA. (*if* $A = \{\}$ *then* *0 else if* $A \in -$ *sigma-Mx* X *then 0 else* ∞))

lemma *measure-space-L*: *measure-space* (*qbs-space* X) (*sigma-Mx* X) (λA. (*if* $A = \{\}$ *then 0 else if* $A \in -$ *sigma-Mx* X *then 0 else* ∞))
 ⟨*proof*⟩

lemma
 shows *space-L*: *space* (*qbs-to-measure* X) = *qbs-space* X (**is** *?goal1*)
  and *sets-L*: *sets* (*qbs-to-measure* X) = *sigma-Mx* X (**is** *?goal2*)
  and *emeasure-L*: *emeasure* (*qbs-to-measure* X) = (λA. *if* $A = \{\} \vee A \notin$ *sigma-Mx* X *then 0 else* ∞) (**is** *?goal3*)
⟨*proof*⟩

lemma *qbs-Mx-sigma-Mx-contra*:
 assumes *qbs-space* X = *qbs-space* Y
  and *qbs-Mx* $X \subseteq$ *qbs-Mx* Y
 shows *sigma-Mx* $Y \subseteq$ *sigma-Mx* X
 ⟨*proof*⟩

The following lemma says that *qbs-to-measure* is a functor from **QBS** to **Meas**.

lemma *l-preserves-morphisms*:
$X \rightarrow_Q Y \subseteq$ (*qbs-to-measure* X) $\rightarrow_M$ (*qbs-to-measure* Y)
⟨*proof*⟩

lemma *qbs-morphism-imp-measurable*: $f \in X \rightarrow_Q Y \Longrightarrow f \in$ *qbs-to-measure* X $\rightarrow_M$ *qbs-to-measure* Y
⟨*proof*⟩

abbreviation *qbs-borel* ($borel_Q$) **where** $borel_Q \equiv$ *measure-to-qbs borel*
abbreviation *qbs-count-space* ($count'\text{-}space_Q$) **where** *qbs-count-space* $I \equiv$ *measure-to-qbs* (*count-space* I)

lemma
shows *qbs-space-qbs-borel*[*simp*]: *qbs-space* $borel_Q$ = *UNIV*
and *qbs-space-count-space*[*simp*]: *qbs-space* (*qbs-count-space* I) = I
and *qbs-Mx-qbs-borel*: *qbs-Mx* $borel_Q$ = *borel-measurable borel*
and *qbs-Mx-count-space*: *qbs-Mx* (*qbs-count-space* I) = *borel* $\rightarrow_M$ *count-space* I
⟨*proof*⟩

lemma
shows *qbs-space-qbs-borel'*[*qbs*]: $r \in$ *qbs-space* $borel_Q$
and *qbs-space-count-space-UNIV'*[*qbs*]: $x \in$ *qbs-space* (*qbs-count-space* (*UNIV* :: (- :: *countable*) *set*))
⟨*proof*⟩

lemma *qbs-Mx-is-morphisms*: *qbs-Mx* X = $borel_Q \rightarrow_Q X$
⟨*proof*⟩

lemma *exp-qbs-Mx'*: *qbs-Mx* (*exp-qbs* X Y) = $\{g.$ *case-prod* $g \in borel_Q \bigotimes_Q X \rightarrow_Q Y\}$
⟨*proof*⟩

lemma *arg-swap-morphism'*:
assumes ($\lambda g.\ f\ (\lambda w\ x.\ g\ x\ w)$) $\in$ *exp-qbs* X (*exp-qbs* W Y) $\rightarrow_Q Z$
shows $f \in$ *exp-qbs* W (*exp-qbs* X Y) $\rightarrow_Q Z$
⟨*proof*⟩

lemma *qbs-Mx-subset-of-measurable*: *qbs-Mx* $X \subseteq$ *borel* $\rightarrow_M$ *qbs-to-measure* X
⟨*proof*⟩

lemma *L-max-of-measurables*:
assumes *space* M = *qbs-space* X
and *qbs-Mx* $X \subseteq$ *borel* $\rightarrow_M M$
shows *sets* $M \subseteq$ *sets* (*qbs-to-measure* X)
⟨*proof*⟩

lemma *qbs-Mx-are-measurable*[*simp*,*measurable*]:
assumes $\alpha \in$ *qbs-Mx X*
shows $\alpha \in$ *borel* $\rightarrow_M$ *qbs-to-measure X*
⟨*proof*⟩

lemma *measure-to-qbs-cong-sets*:
assumes *sets M* = *sets N*
shows *measure-to-qbs M* = *measure-to-qbs N*
⟨*proof*⟩

lemma *lr-sets*[*simp*]:
sets X ⊆ *sets* (*qbs-to-measure* (*measure-to-qbs X*))
⟨*proof*⟩

lemma(**in** *standard-borel*) *lr-sets-ident*[*simp*, *measurable-cong*]:
sets (*qbs-to-measure* (*measure-to-qbs M*)) = *sets M*
⟨*proof*⟩

corollary *sets-lr-polish-borel*[*simp*, *measurable-cong*]: *sets* (*qbs-to-measure qbs-borel*) = *sets* (*borel* :: (- :: *polish-space*) *measure*)
⟨*proof*⟩

corollary *sets-lr-count-space*[*simp*, *measurable-cong*]: *sets* (*qbs-to-measure* (*qbs-count-space* (*UNIV* :: (- :: *countable*) *set*))) = *sets* (*count-space UNIV*)
⟨*proof*⟩

lemma *map-qbs-embed-measure1*:
assumes *inj-on f* (*space M*)
shows *map-qbs f* (*measure-to-qbs M*) = *measure-to-qbs* (*embed-measure M f*)
⟨*proof*⟩

lemma *map-qbs-embed-measure2*:
assumes *inj-on f* (*qbs-space X*)
shows *sets* (*qbs-to-measure* (*map-qbs f X*)) = *sets* (*embed-measure* (*qbs-to-measure X*) *f*)
⟨*proof*⟩

lemma(**in** *standard-borel*) *map-qbs-embed-measure2′*:
assumes *inj-on f* (*space M*)
shows *sets* (*qbs-to-measure* (*map-qbs f* (*measure-to-qbs M*))) = *sets* (*embed-measure M f*)
⟨*proof*⟩

3.3.3 The Adjunction

lemma *lr-adjunction-correspondence* :
$X \rightarrow_Q$ (*measure-to-qbs Y*) = (*qbs-to-measure X*) $\rightarrow_M$ Y
⟨*proof*⟩

lemma(**in** *standard-borel*) *standard-borel-r-full-faithful*:
$M \to_M Y = measure\text{-}to\text{-}qbs\ M \to_Q measure\text{-}to\text{-}qbs\ Y$
⟨*proof*⟩

lemma *qbs-morphism-dest*[*measurable-dest*]:
assumes $f \in X \to_Q measure\text{-}to\text{-}qbs\ Y$
shows $f \in qbs\text{-}to\text{-}measure\ X \to_M Y$
⟨*proof*⟩

lemma(**in** *standard-borel*) *qbs-morphism-dest*:
assumes $k \in measure\text{-}to\text{-}qbs\ M \to_Q measure\text{-}to\text{-}qbs\ Y$
shows $k \in M \to_M Y$
⟨*proof*⟩

lemma *qbs-morphism-measurable-intro*:
assumes $f \in qbs\text{-}to\text{-}measure\ X \to_M Y$
shows $f \in X \to_Q measure\text{-}to\text{-}qbs\ Y$
⟨*proof*⟩

lemma(**in** *standard-borel*) *qbs-morphism-measurable-intro*:
assumes $k \in M \to_M Y$
shows $k \in measure\text{-}to\text{-}qbs\ M \to_Q measure\text{-}to\text{-}qbs\ Y$
⟨*proof*⟩

lemma *r-preserves-product* :
$measure\text{-}to\text{-}qbs\ (X \bigotimes_M Y) = measure\text{-}to\text{-}qbs\ X \bigotimes_Q measure\text{-}to\text{-}qbs\ Y$
⟨*proof*⟩

lemma *l-product-sets*:
$sets\ (qbs\text{-}to\text{-}measure\ X \bigotimes_M qbs\text{-}to\text{-}measure\ Y) \subseteq sets\ (qbs\text{-}to\text{-}measure\ (X \bigotimes_Q Y))$
⟨*proof*⟩

corollary *qbs-borel-prod*: $qbs\text{-}borel \bigotimes_Q qbs\text{-}borel = (qbs\text{-}borel :: ('a::second\text{-}countable\text{-}topology \times 'b::second\text{-}countable\text{-}topology)\ quasi\text{-}borel)$
⟨*proof*⟩

corollary *qbs-count-space-prod*: $qbs\text{-}count\text{-}space\ (UNIV :: ('a :: countable)\ set) \bigotimes_Q qbs\text{-}count\text{-}space\ (UNIV :: ('b :: countable)\ set) = qbs\text{-}count\text{-}space\ UNIV$
⟨*proof*⟩

lemma *r-preserves-product'*: $measure\text{-}to\text{-}qbs\ (\Pi_M\ i{\in}I.\ M\ i) = (\Pi_Q\ i{\in}I.\ measure\text{-}to\text{-}qbs\ (M\ i))$
⟨*proof*⟩

lemma *PiQ-qbs-borel*:
$(\Pi_Q\ i{::}('a{::}\ countable){\in}UNIV.\ (qbs\text{-}borel :: ('b{::}second\text{-}countable\text{-}topology\ quasi\text{-}borel))) = qbs\text{-}borel$
⟨*proof*⟩

lemma *qbs-morphism-from-countable*:
fixes X :: *'a quasi-borel*
assumes *countable* (*qbs-space* X)
qbs-Mx $X \subseteq$ *borel* $\rightarrow_M$ *count-space* (*qbs-space* X)
and $\bigwedge i.\ i \in$ *qbs-space* $X \Longrightarrow f\ i \in$ *qbs-space* Y
shows $f \in X \rightarrow_Q Y$
⟨*proof*⟩

corollary *qbs-morphism-count-space'*:
assumes $\bigwedge i.\ i \in I \Longrightarrow f\ i \in$ *qbs-space* Y *countable* I
shows $f \in$ *qbs-count-space* $I \rightarrow_Q Y$
⟨*proof*⟩

corollary *qbs-morphism-count-space*:
assumes $\bigwedge i.\ f\ i \in$ *qbs-space* Y
shows $f \in$ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q Y$
⟨*proof*⟩

lemma [*qbs*]:
shows *not-qbs-pred*: *Not* ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *qbs-count-space UNIV*
and *or-qbs-pred*: (∨) ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *and-qbs-pred*: (∧) ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *implies-qbs-pred*: (⟶) ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *iff-qbs-pred*: (⟷) ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
⟨*proof*⟩

lemma [*qbs*]:
shows *less-count-qbs-pred*: (<) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *le-count-qbs-pred*: (≤) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *eq-count-qbs-pred*: (=) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *plus-count-qbs-morphism*: (+) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *minus-count-qbs-morphism*: (−) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *mult-count-qbs-morphism*: (∗) ∈ *qbs-count-space* (*UNIV* :: (- :: *countable*) *set*) $\rightarrow_Q$ *exp-qbs* (*qbs-count-space UNIV*) (*qbs-count-space UNIV*)
and *Suc-qbs-morphism*: *Suc* ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *qbs-count-space UNIV*
⟨*proof*⟩

lemma *qbs-morphism-product-iff*:
$f \in X \rightarrow_Q (\Pi_Q\ i$ :: (- :: *countable*)∈*UNIV*. $Y) \longleftrightarrow f \in X \rightarrow_Q$ *qbs-count-space*

UNIV $\Rightarrow_Q$ *Y*
⟨*proof*⟩

lemma *qbs-morphism-pair-countable1*:
  assumes *countable* (*qbs-space X*)
          qbs-Mx X ⊆ *borel* $\rightarrow_M$ *count-space* (*qbs-space X*)
      and ⋀*i*. *i* ∈ *qbs-space X* ⟹ *f i* ∈ *Y* $\rightarrow_Q$ *Z*
    shows (λ(*x*,*y*). *f x y*) ∈ *X* $\bigotimes_Q$ *Y* $\rightarrow_Q$ *Z*
  ⟨*proof*⟩

lemma *qbs-morphism-pair-countable2*:
  assumes *countable* (*qbs-space Y*)
          qbs-Mx Y ⊆ *borel* $\rightarrow_M$ *count-space* (*qbs-space Y*)
      and ⋀*i*. *i* ∈ *qbs-space Y* ⟹ (λ*x*. *f x i*) ∈ *X* $\rightarrow_Q$ *Z*
    shows (λ(*x*,*y*). *f x y*) ∈ *X* $\bigotimes_Q$ *Y* $\rightarrow_Q$ *Z*
  ⟨*proof*⟩

corollary *qbs-morphism-pair-count-space1*:
  assumes ⋀*i*. *f i* ∈ *Y* $\rightarrow_Q$ *Z*
  shows (λ(*x*,*y*). *f x y*) ∈ *qbs-count-space* (*UNIV* :: (*'a* :: *countable*) *set*) $\bigotimes_Q$ *Y* $\rightarrow_Q$ *Z*
  ⟨*proof*⟩

corollary *qbs-morphism-pair-count-space2*:
  assumes ⋀*i*. (λ*x*. *f x i*) ∈ *X* $\rightarrow_Q$ *Z*
  shows (λ(*x*,*y*). *f x y*) ∈ *X* $\bigotimes_Q$ *qbs-count-space* (*UNIV* :: (*'a* :: *countable*) *set*) $\rightarrow_Q$ *Z*
  ⟨*proof*⟩

lemma *qbs-morphism-compose-countable'*:
  assumes [*qbs*]:⋀*i*. *i* ∈ *I* ⟹ (λ*x*. *f i x*) ∈ *X* $\rightarrow_Q$ *Y* *g* ∈ *X* $\rightarrow_Q$ *qbs-count-space I* *countable I*
  shows (λ*x*. *f* (*g x*) *x*) ∈ *X* $\rightarrow_Q$ *Y*
⟨*proof*⟩

lemma *qbs-morphism-compose-countable*:
 assumes [*simp*]:⋀*i*::*'i*::*countable*. (λ*x*. *f i x*) ∈ *X* $\rightarrow_Q$ *Y* *g* ∈ *X* $\rightarrow_Q$ (*qbs-count-space UNIV*)
  shows (λ*x*. *f* (*g x*) *x*) ∈ *X* $\rightarrow_Q$ *Y*
  ⟨*proof*⟩

lemma *qbs-morphism-op*:
  assumes *case-prod f* ∈ *X* $\bigotimes_M$ *Y* $\rightarrow_M$ *Z*
  shows *f* ∈ *measure-to-qbs X* $\rightarrow_Q$ *measure-to-qbs Y* $\Rightarrow_Q$ *measure-to-qbs Z*
  ⟨*proof*⟩

lemma [*qbs*]:
 shows *plus-qbs-morphism*: (+) ∈ (*qbs-borel* :: (-::{*second-countable-topology*, *topological-monoid-add*}) *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*

and *plus-ereal-qbs-morphism*: $(+) \in$ (*qbs-borel* :: *ereal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *diff-qbs-morphism*: $(-) \in$ (*qbs-borel* :: (-::{*second-countable-topology*, *real-normed-vector*}) *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *diff-ennreal-qbs-morphism*: $(-) \in$ (*qbs-borel* :: *ennreal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *diff-ereal-qbs-morphism*: $(-) \in$ (*qbs-borel* :: *ereal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *times-qbs-morphism*: $(*) \in$ (*qbs-borel* :: (-::{*second-countable-topology*, *real-normed-algebra*}) *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *times-ennreal-qbs-morphism*: $(*) \in$ (*qbs-borel* :: *ennreal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *times-ereal-qbs-morphism*: $(*) \in$ (*qbs-borel* :: *ereal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *divide-qbs-morphism*: $(/) \in$ (*qbs-borel* :: (-::{*second-countable-topology*, *real-normed-div-algebra*}) *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *divide-ennreal-qbs-morphism*: $(/) \in$ (*qbs-borel* :: *ennreal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *divide-ereal-qbs-morphism*: $(/) \in$ (*qbs-borel* :: *ereal quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *log-qbs-morphism*: *log* $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *root-qbs-morphism*: *root* $\in$ *qbs-count-space UNIV* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *scaleR-qbs-morphism*: $(*_R) \in$ *qbs-borel* $\rightarrow_Q$ (*qbs-borel* :: (-::{*second-countable-topology*, *real-normed-vector*}) *quasi-borel*) $\Rightarrow_Q$ *qbs-borel*
and *qbs-morphism-inner*: $(\cdot) \in$ *qbs-borel* $\rightarrow_Q$ (*qbs-borel* :: (-::{*second-countable-topology*, *real-inner*}) *quasi-borel*) $\Rightarrow_Q$ *qbs-borel*
and *dist-qbs-morphism*: *dist* $\in$ (*qbs-borel* :: (-::{*second-countable-topology*, *metric-space*}) *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *powr-qbs-morphism*: (*powr*) $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ (*qbs-borel* :: *real quasi-borel*)
and *max-qbs-morphism*: (*max* :: (- :: {*second-countable-topology*, *linorder-topology*}) $\Rightarrow$ - $\Rightarrow$ -) $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *min-qbs-morphism*: (*min* :: (- :: {*second-countable-topology*, *linorder-topology*}) $\Rightarrow$ - $\Rightarrow$ -) $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *sup-qbs-morphism*: (*sup* :: (- :: {*lattice*,*second-countable-topology*, *linorder-topology*}) $\Rightarrow$ - $\Rightarrow$ -) $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *inf-qbs-morphism*: (*inf* :: (- :: {*lattice*,*second-countable-topology*, *linorder-topology*}) $\Rightarrow$ - $\Rightarrow$ -) $\in$ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
and *less-qbs-pred*: $(<) \in$ (*qbs-borel* :: - ::{*second-countable-topology*, *linorder-topology*} *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-count-space UNIV*
and *eq-qbs-pred*: $(=) \in$ (*qbs-borel* :: - ::{*second-countable-topology*, *linorder-topology*} *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-count-space UNIV*
and *le-qbs-pred*: $(\leq) \in$ (*qbs-borel* :: - ::{*second-countable-topology*, *linorder-topology*} *quasi-borel*) $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-count-space UNIV*
⟨*proof*⟩

lemma [*qbs*]:
shows *abs-real-qbs-morphism*: *abs* $\in$ (*qbs-borel* :: *real quasi-borel*) $\rightarrow_Q$ *qbs-borel*
and *abs-ereal-qbs-morphism*: *abs* $\in$ (*qbs-borel* :: *ereal quasi-borel*) $\rightarrow_Q$ *qbs-borel*

and *real-floor-qbs-morphism*: $(floor :: real \Rightarrow int) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}count\text{-}space$ *UNIV*
and *real-ceiling-qbs-morphism*: $(ceiling :: real \Rightarrow int) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}count\text{-}space$ *UNIV*
and *exp-qbs-morphism*: $(exp::'a::\{real\text{-}normed\text{-}field,banach\}\Rightarrow'a) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *ln-qbs-morphism*: $ln \in (qbs\text{-}borel :: real\ quasi\text{-}borel) \rightarrow_Q qbs\text{-}borel$
and *sqrt-qbs-morphism*: $sqrt \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *of-real-qbs-morphism*: $(of\text{-}real :: - \Rightarrow (-::real\text{-}normed\text{-}algebra)) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *sin-qbs-morphism*: $(sin :: - \Rightarrow (-::\{real\text{-}normed\text{-}field,banach\})) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *cos-qbs-morphism*: $(cos :: - \Rightarrow (-::\{real\text{-}normed\text{-}field,banach\})) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *arctan-qbs-morphism*: $arctan \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *Re-qbs-morphism*: $Re \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *Im-qbs-morphism*: $Im \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *sgn-qbs-morphism*: $(sgn::-::real\text{-}normed\text{-}vector \Rightarrow -) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *norm-qbs-morphism*: $norm \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *invers-qbs-morphism*: $(inverse :: - \Rightarrow (- ::real\text{-}normed\text{-}div\text{-}algebra)) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *invers-ennreal-qbs-morphism*: $(inverse :: - \Rightarrow ennreal) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *invers-ereal-qbs-morphism*: $(inverse :: - \Rightarrow ereal) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *uminus-qbs-morphism*: $(uminus :: - \Rightarrow (-::\{second\text{-}countable\text{-}topology, real\text{-}normed\text{-}vector\})) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *ereal-qbs-morphism*: $ereal \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *real-of-ereal-qbs-morphism*: $real\text{-}of\text{-}ereal \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *enn2ereal-qbs-morphism*: $enn2ereal \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *e2ennreal-qbs-morphism*: $e2ennreal \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *ennreal-qbs-morphism*: $ennreal \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *qbs-morphism-nth*: $(\lambda x::real\hat{}'n.\ x \$ i) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *qbs-morphism-product-candidate*: $\bigwedge i.\ (\lambda x.\ x\ i) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
and *uminus-ereal-qbs-morphism*: $(uminus :: - \Rightarrow ereal) \in qbs\text{-}borel \rightarrow_Q qbs\text{-}borel$
⟨*proof*⟩

lemma *qbs-morphism-sum*:
fixes $f :: 'c \Rightarrow 'a \Rightarrow 'b::\{second\text{-}countable\text{-}topology, topological\text{-}comm\text{-}monoid\text{-}add\}$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \rightarrow_Q qbs\text{-}borel$
shows $(\lambda x.\ \sum i{\in}S.\ f\ i\ x) \in X \rightarrow_Q qbs\text{-}borel$
⟨*proof*⟩

lemma *qbs-morphism-suminf-order*:
fixes $f :: nat \Rightarrow 'a \Rightarrow 'b::\{complete\text{-}linorder, second\text{-}countable\text{-}topology, linorder\text{-}topology, topological\text{-}comm\text{-}monoid\text{-}add\}$
assumes $\bigwedge i.\ f\ i \in X \rightarrow_Q qbs\text{-}borel$
shows $(\lambda x.\ \sum i.\ f\ i\ x) \in X \rightarrow_Q qbs\text{-}borel$
⟨*proof*⟩

lemma *qbs-morphism-prod*:
fixes f :: $'c \Rightarrow 'a \Rightarrow 'b$::{*second-countable-topology*, *real-normed-field*}
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \to_Q$ *qbs-borel*
shows $(\lambda x. \prod i{\in}S.\ f\ i\ x) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-Min*:
finite $I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow f\ i \in X \to_Q$ *qbs-borel*$) \Longrightarrow (\lambda x.\ Min\ ((\lambda i.\ f\ i\ x)\text{`}I)$ :: $'b$::{*second-countable-topology*, *linorder-topology*}$) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-Max*:
finite $I \Longrightarrow (\bigwedge i.\ i \in I \Longrightarrow f\ i \in X \to_Q$ *qbs-borel*$) \Longrightarrow (\lambda x.\ Max\ ((\lambda i.\ f\ i\ x)\text{`}I)$ :: $'b$::{*second-countable-topology*, *linorder-topology*}$) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-Max2*:
fixes f::- $\Rightarrow$ - $\Rightarrow$ $'a$::{*second-countable-topology*, *dense-linorder*, *linorder-topology*}
shows *finite* $I \Longrightarrow (\bigwedge i.\ f\ i \in X \to_Q$ *qbs-borel*$) \Longrightarrow (\lambda x.\ Max\{f\ i\ x\ |i.\ i \in I\}) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma [*qbs*]:
shows *qbs-morphism-liminf*: *liminf* $\in$ (*qbs-count-space UNIV* $\Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ (*qbs-borel* :: $'a$ :: {*complete-linorder*, *second-countable-topology*, *linorder-topology*} *quasi-borel*)
and *qbs-morphism-limsup*: *limsup* $\in$ (*qbs-count-space UNIV* $\Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ (*qbs-borel* :: $'a$ :: {*complete-linorder*, *second-countable-topology*, *linorder-topology*} *quasi-borel*)
and *qbs-morphism-lim*: *lim* $\in$ (*qbs-count-space UNIV* $\Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ (*qbs-borel* :: $'a$ :: {*complete-linorder*, *second-countable-topology*, *linorder-topology*} *quasi-borel*)
⟨*proof*⟩

lemma *qbs-morphism-SUP*:
fixes F :: - $\Rightarrow$ - $\Rightarrow$ -::{*complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes *countable* I $\bigwedge i.\ i \in I \Longrightarrow F\ i \in X \to_Q$ *qbs-borel*
shows $(\lambda x. \bigsqcup i{\in}I.\ F\ i\ x) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-INF*:
fixes F :: - $\Rightarrow$ - $\Rightarrow$ -::{*complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes *countable* I $\bigwedge i.\ i \in I \Longrightarrow F\ i \in X \to_Q$ *qbs-borel*
shows $(\lambda x. \bigsqcap i{\in}I.\ F\ i\ x) \in X \to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-cSUP*:
fixes F :: - $\Rightarrow$ - $\Rightarrow$ $'a$::{*conditionally-complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes *countable* I $\bigwedge i.\ i \in I \Longrightarrow F\ i \in X \to_Q$ *qbs-borel* $\bigwedge x.\ x \in$ *qbs-space* X

$\Longrightarrow$ *bdd-above* ((λ*i*. *F i x*) ‘ *I*)
shows (λ*x*. $\bigsqcup$ *i*∈*I*. *F i x*) ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-cINF*:
fixes *F* :: - ⇒ - ⇒ ′*a*::{*conditionally-complete-linorder*, *linorder-topology*, *second-countable-topology*}
assumes *countable I* ⋀*i*. *i* ∈ *I* $\Longrightarrow$ *F i* ∈ *X* $\to_Q$ *qbs-borel* ⋀*x*. *x* ∈ *qbs-space X* $\Longrightarrow$ *bdd-below* ((λ*i*. *F i x*) ‘ *I*)
shows (λ*x*. $\bigsqcap$ *i*∈*I*. *F i x*) ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-lim-metric*:
fixes *f* :: *nat* ⇒ ′*a* ⇒ ′*b*::{*banach*, *second-countable-topology*}
assumes ⋀*i*. *f i* ∈ *X* $\to_Q$ *qbs-borel*
shows (λ*x*. *lim* (λ*i*. *f i x*)) ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-LIMSEQ-metric*:
fixes *f* :: *nat* ⇒ ′*a* ⇒ ′*b* :: *metric-space*
assumes ⋀*i*. *f i* ∈ *X* $\to_Q$ *qbs-borel* ⋀*x*. *x* ∈ *qbs-space X* $\Longrightarrow$ (λ*i*. *f i x*) $\longrightarrow$ *g x*
shows *g* ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *power-qbs-morphism*[*qbs*]:
(*power* :: (- ::{*power*,*real-normed-algebra*}) ⇒ *nat* ⇒ -) ∈ *qbs-borel* $\to_Q$ *qbs-count-space UNIV* $\Rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *power-ennreal-qbs-morphism*[*qbs*]:
(*power* :: *ennreal* ⇒ *nat* ⇒ -) ∈ *qbs-borel* $\to_Q$ *qbs-count-space UNIV* $\Rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-morphism-compw*: ($\frown\frown$) ∈ (*X* $\Rightarrow_Q$ *X*) $\to_Q$ *qbs-count-space UNIV* $\Rightarrow_Q$ (*X* $\Rightarrow_Q$ *X*)
⟨*proof*⟩

lemma *qbs-morphism-compose-n*[*qbs*]:
assumes [*qbs*]: *f* ∈ *X* $\to_Q$ *X*
shows (λ*n*. *f* $\frown\frown$ *n*) ∈ *qbs-count-space UNIV* $\to_Q$ *X* $\Rightarrow_Q$ *X*
⟨*proof*⟩

lemma *qbs-morphism-compose-n*′:
assumes *f* ∈ *X* $\to_Q$ *X*
shows *f* $\frown\frown$ *n* ∈ *X* $\to_Q$ *X*
⟨*proof*⟩

lemma *qbs-morphism-uminus-eq-ereal*[*simp*]:

$(\lambda x. - f\ x :: ereal) \in X \to_Q qbs\text{-}borel \longleftrightarrow f \in X \to_Q qbs\text{-}borel$ (**is** $?l = ?r$)
$\langle proof \rangle$

lemma *qbs-morphism-ereal-iff*:
shows $(\lambda x.\ ereal\ (f\ x)) \in X \to_Q qbs\text{-}borel \longleftrightarrow f \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *qbs-morphism-ereal-sum*:
fixes $f :: 'c \Rightarrow 'a \Rightarrow ereal$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \to_Q qbs\text{-}borel$
shows $(\lambda x. \sum i{\in}S.\ f\ i\ x) \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *qbs-morphism-ereal-prod*:
fixes $f :: 'c \Rightarrow 'a \Rightarrow ereal$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \to_Q qbs\text{-}borel$
shows $(\lambda x. \prod i{\in}S.\ f\ i\ x) \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *qbs-morphism-extreal-suminf*:
fixes $f :: nat \Rightarrow 'a \Rightarrow ereal$
assumes $\bigwedge i.\ f\ i \in X \to_Q qbs\text{-}borel$
shows $(\lambda x. (\sum i.\ f\ i\ x)) \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *qbs-morphism-ennreal-iff*:
assumes $\bigwedge x.\ x \in qbs\text{-}space\ X \Longrightarrow 0 \le f\ x$
shows $(\lambda x.\ ennreal\ (f\ x)) \in X \to_Q qbs\text{-}borel \longleftrightarrow f \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *qbs-morphism-prod-ennreal*:
fixes $f :: 'c \Rightarrow 'a \Rightarrow ennreal$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \to_Q qbs\text{-}borel$
shows $(\lambda x. \prod i{\in}S.\ f\ i\ x) \in X \to_Q qbs\text{-}borel$
$\langle proof \rangle$

lemma *count-space-qbs-morphism*:
$f \in qbs\text{-}count\text{-}space\ (UNIV :: 'a\ set) \to_Q qbs\text{-}borel$
$\langle proof \rangle$

declare *count-space-qbs-morphism*[**where** $'a$=- :: *countable,qbs*]

lemma *count-space-count-space-qbs-morphism*:
$f \in qbs\text{-}count\text{-}space\ (UNIV :: (- :: countable)\ set) \to_Q qbs\text{-}count\text{-}space\ (UNIV :: (- :: countable)\ set)$
$\langle proof \rangle$

lemma *qbs-morphism-case-nat'*:
assumes [*qbs*]: $i = 0 \Longrightarrow f \in X \to_Q Y$

$\bigwedge j.\ i = Suc\ j \Longrightarrow (\lambda x.\ g\ x\ j) \in X \rightarrow_Q Y$
shows $(\lambda x.\ case\text{-}nat\ (f\ x)\ (g\ x)\ i) \in X \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-case-nat*[*qbs*]:
$case\text{-}nat \in X \rightarrow_Q (qbs\text{-}count\text{-}space\ UNIV \Rightarrow_Q X) \Rightarrow_Q qbs\text{-}count\text{-}space\ UNIV \Rightarrow_Q X$
⟨*proof*⟩

lemma *qbs-morphism-case-nat''*:
assumes $f \in X \rightarrow_Q Y\ g \in X \rightarrow_Q (\Pi_Q\ i{\in}UNIV.\ Y)$
shows $(\lambda x.\ case\text{-}nat\ (f\ x)\ (g\ x)) \in X \rightarrow_Q (\Pi_Q\ i{\in}UNIV.\ Y)$
⟨*proof*⟩

lemma *qbs-morphism-rec-nat*[*qbs*]: $rec\text{-}nat \in X \rightarrow_Q (count\text{-}space\ UNIV \Rightarrow_Q X \Rightarrow_Q X) \Rightarrow_Q count\text{-}space\ UNIV \Rightarrow_Q X$
⟨*proof*⟩

lemma *qbs-morphism-Max-nat*:
fixes $P :: nat \Rightarrow 'a \Rightarrow bool$
assumes $\bigwedge i.\ P\ i \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
shows $(\lambda x.\ Max\ \{i.\ P\ i\ x\}) \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
⟨*proof*⟩

lemma *qbs-morphism-Min-nat*:
fixes $P :: nat \Rightarrow 'a \Rightarrow bool$
assumes $\bigwedge i.\ P\ i \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
shows $(\lambda x.\ Min\ \{i.\ P\ i\ x\}) \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
⟨*proof*⟩

lemma *qbs-morphism-sum-nat*:
fixes $f :: 'c \Rightarrow 'a \Rightarrow nat$
assumes $\bigwedge i.\ i \in S \Longrightarrow f\ i \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
shows $(\lambda x.\ \sum i{\in}S.\ f\ i\ x) \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$
⟨*proof*⟩

lemma *qbs-morphism-case-enat'*:
assumes *f*[*qbs*]: $f \in X \rightarrow_Q qbs\text{-}count\text{-}space\ UNIV$ **and** [*qbs*]: $\bigwedge i.\ g\ i \in X \rightarrow_Q Y\ h \in X \rightarrow_Q Y$
shows $(\lambda x.\ case\ f\ x\ of\ enat\ i \Rightarrow g\ i\ x \mid \infty \Rightarrow h\ x) \in X \rightarrow_Q Y$
⟨*proof*⟩

lemma *qbs-morphism-case-enat*[*qbs*]: $case\text{-}enat \in qbs\text{-}space\ ((qbs\text{-}count\text{-}space\ UNIV \Rightarrow_Q X) \Rightarrow_Q X \Rightarrow_Q qbs\text{-}count\text{-}space\ UNIV \Rightarrow_Q X)$
⟨*proof*⟩

lemma *qbs-morphism-restrict*[*qbs*]:

assumes X: $\bigwedge i.\ i \in I \Longrightarrow f\ i \in X \rightarrow_Q (Y\ i)$
shows $(\lambda x.\ \lambda i{\in}I.\ f\ i\ x) \in X \rightarrow_Q (\Pi_Q\ i{\in}I.\ Y\ i)$
⟨*proof*⟩

lemma *If-qbs-morphism*[*qbs*]: *If* ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *X* $\Rightarrow_Q$ *X* $\Rightarrow_Q$ *X*
⟨*proof*⟩

lemma *normal-density-qbs*[*qbs*]: *normal-density* ∈ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *erlang-density-qbs*[*qbs*]: *erlang-density* ∈ *qbs-count-space UNIV* $\rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel* $\Rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *list-nil-qbs*[*qbs*]: [] ∈ *qbs-space* (*list-qbs X*)
⟨*proof*⟩

lemma *list-cons-qbs-morphism*: *list-cons* ∈ $X \rightarrow_Q (\Pi_Q\ n{\in}(UNIV :: nat\ set).\Pi_Q\ i{\in}\{..{<}n\}.\ X) \Rightarrow_Q (\Pi_Q\ n{\in}(UNIV :: nat\ set).\Pi_Q\ i{\in}\{..{<}n\}.\ X)$
⟨*proof*⟩

corollary *cons-qbs-morphism*[*qbs*]: *Cons* ∈ *X* $\rightarrow_Q$ (*list-qbs X*) $\Rightarrow_Q$ *list-qbs X*
⟨*proof*⟩

lemma *rec-list-morphism′*:
rec-list′ ∈ *qbs-space* $(Y \Rightarrow_Q (X \Rightarrow_Q (\Pi_Q\ n{\in}(UNIV :: nat\ set).\Pi_Q\ i{\in}\{..{<}n\}.\ X) \Rightarrow_Q Y \Rightarrow_Q Y) \Rightarrow_Q (\Pi_Q\ n{\in}(UNIV :: nat\ set).\Pi_Q\ i{\in}\{..{<}n\}.\ X) \Rightarrow_Q Y)$
⟨*proof*⟩

lemma *rec-list-morphism*[*qbs*]: *rec-list* ∈ *qbs-space* (*Y* $\Rightarrow_Q$ (*X* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *Y*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y*)
⟨*proof*⟩

hide-const (**open**) *list-nil list-cons list-head list-tail from-list rec-list′ to-list′*

hide-fact (**open**) *list-simp1 list-simp2 list-simp3 list-simp4 list-simp5 list-simp6 list-simp7 from-list-in-list-of′ list-cons-qbs-morphism rec-list′-simp1 to-list-from-list-ident from-list-in-list-of to-list-set to-list-simp1 to-list-simp2 list-head-def list-tail-def from-list-length list-cons-in-list-of rec-list-morphism′ rec-list′-simp2 list-decomp1 list-destruct-rule list-induct-rule from-list-to-list-ident*

corollary *case-list-morphism*[*qbs*]: *case-list* ∈ *qbs-space* ((*Y* :: ′*b quasi-borel*) $\Rightarrow_Q$ ((*X* :: ′*a quasi-borel*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y*)
⟨*proof*⟩

lemma *fold-qbs-morphism*[*qbs*]: *fold* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *Y*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *Y*)

⟨proof⟩

lemma [*qbs*]:
shows *foldr-qbs-morphism*: *foldr* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *Y*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *Y*)
and *foldl-qbs-morphism*: *foldl* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *Y* $\Rightarrow_Q$ *X*) $\Rightarrow_Q$ *X* $\Rightarrow_Q$ *list-qbs Y* $\Rightarrow_Q$ *X*)
and *zip-qbs-morphism*: *zip* ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *list-qbs Y* $\Rightarrow_Q$ *list-qbs* (*pair-qbs X Y*))
and *append-qbs-morphism*: *append* ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
and *concat-qbs-morphism*: *concat* ∈ *qbs-space* (*list-qbs* (*list-qbs X*) $\Rightarrow_Q$ *list-qbs X*)
and *drop-qbs-morphism*: *drop* ∈ *qbs-space* (*qbs-count-space UNIV* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
and *take-qbs-morphism*: *take* ∈ *qbs-space* (*qbs-count-space UNIV* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
and *rev-qbs-morphism*: *rev* ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
⟨proof⟩

lemma [*qbs*]:
fixes *X* :: *'a quasi-borel* **and** *Y* :: *'b quasi-borel*
shows *map-qbs-morphism*: *map* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *Y*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs Y*) (**is** *?map*)
and *fileter-qbs-morphism*: *filter* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *count-space*$_Q$ *UNIV*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*) (**is** *?filter*)
and *length-qbs-morphism*: *length* ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *qbs-count-space UNIV*) (**is** *?length*)
and *tl-qbs-morphism*: *tl* ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *list-qbs X*) (**is** *?tl*)
and *list-all-qbs-morphism*: *list-all* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *qbs-count-space UNIV*) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *qbs-count-space UNIV*) (**is** *?list-all*)
and *bind-list-qbs-morphism*: (⤜) ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ (*X* $\Rightarrow_Q$ *list-qbs Y*) $\Rightarrow_Q$ *list-qbs Y*) (**is** *?bind*)
⟨proof⟩

lemma *list-eq-qbs-morphism*[*qbs*]:
assumes [*qbs*]: (=) ∈ *qbs-space* (*X* $\Rightarrow_Q$ *X* $\Rightarrow_Q$ *count-space UNIV*)
shows (=) ∈ *qbs-space* (*list-qbs X* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *count-space UNIV*)
⟨proof⟩

lemma *insort-key-qbs-morphism*[*qbs*]:
shows *insort-key* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ (*borel*$_Q$ ::'*b* :: {*second-countable-topology*, *linorder-topology*} *quasi-borel*)) $\Rightarrow_Q$ *X* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*) (**is** *?g1*)
and *insort-key* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *count-space*$_Q$ (*UNIV* :: (- :: *countable*) *set*)) $\Rightarrow_Q$ *X* $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*) (**is** *?g2*)
⟨proof⟩

lemma *sort-key-qbs-morphism*[*qbs*]:
shows *sort-key* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ (*borel*$_Q$ ::'*b* :: {*second-countable-topology*,

linorder-topology} *quasi-borel*)) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
and *sort-key* ∈ *qbs-space* ((*X* $\Rightarrow_Q$ *count-space*$_Q$ (*UNIV* :: (- :: *countable*) *set*)) $\Rightarrow_Q$ *list-qbs X* $\Rightarrow_Q$ *list-qbs X*)
⟨*proof*⟩

lemma *sort-qbs-morphism*[*qbs*]:
shows *sort* ∈ *list-qbs* (*borel*$_Q$ ::'*b* :: {*second-countable-topology*, *linorder-topology*} *quasi-borel*) $\rightarrow_Q$ *list-qbs borel*$_Q$
and *sort* ∈ *list-qbs* (*count-space*$_Q$ (*UNIV* :: (- :: *countable*) *set*)) $\rightarrow_Q$ *list-qbs* (*count-space*$_Q$ *UNIV*)
⟨*proof*⟩

3.3.4 Morphism Pred

abbreviation *qbs-pred X P* ≡ *P* ∈ *X* $\rightarrow_Q$ *qbs-count-space* (*UNIV* :: *bool set*)

lemma *qbs-pred-iff-measurable-pred*:
qbs-pred X P = *Measurable.pred* (*qbs-to-measure X*) *P*
⟨*proof*⟩

lemma(**in** *standard-borel*) *qbs-pred-iff-measurable-pred*:
qbs-pred (*measure-to-qbs M*) *P* = *Measurable.pred M P*
⟨*proof*⟩

lemma *qbs-pred-iff-sets*:
{*x* ∈*space* (*qbs-to-measure X*). *P x*} ∈ *sets* (*qbs-to-measure X*) ⟷ *qbs-pred X P*
⟨*proof*⟩

lemma
assumes [*qbs*]:*P* ∈ *X* $\rightarrow_Q$ *Y* $\Rightarrow_Q$ *qbs-count-space UNIV f* ∈ *X* $\rightarrow_Q$ *Y*
shows *indicator-qbs-morphism'''*: (λ*x*. *indicator* {*y*. *P x y*} (*f x*)) ∈ *X* $\rightarrow_Q$ *qbs-borel* (**is** *?g1*)
and *indicator-qbs-morphism''*: (λ*x*. *indicator* {*y*∈*qbs-space Y*. *P x y*} (*f x*)) ∈ *X* $\rightarrow_Q$ *qbs-borel* (**is** *?g2*)
⟨*proof*⟩

lemma
assumes [*qbs*]:*P* ∈ *X* $\rightarrow_Q$ *Y* $\Rightarrow_Q$ *qbs-count-space UNIV*
shows *indicator-qbs-morphism*[*qbs*]:(λ*x*. *indicator* {*y* ∈ *qbs-space Y*. *P x y*}) ∈ *X* $\rightarrow_Q$ *Y* $\Rightarrow_Q$ *qbs-borel* (**is** *?g1*)
and *indicator-qbs-morphism'*:(λ*x*. *indicator* {*y*. *P x y*}) ∈ *X* $\rightarrow_Q$ *Y* $\Rightarrow_Q$ *qbs-borel* (**is** *?g2*)
⟨*proof*⟩

lemma *indicator-qbs*[*qbs*]:
assumes *qbs-pred X P*
shows *indicator* {*x*. *P x*} ∈ *X* $\rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *All-qbs-pred*[*qbs*]: *qbs-pred* (*count-space*$_Q$ (*UNIV* :: ('*a* :: *countable*) *set*) $\Rightarrow_Q$ *count-space*$_Q$ *UNIV*) *All*
⟨*proof*⟩

lemma *Ex-qbs-pred*[*qbs*]: *qbs-pred* (*count-space*$_Q$ (*UNIV* :: ('*a* :: *countable*) *set*) $\Rightarrow_Q$ *count-space*$_Q$ *UNIV*) *Ex*
⟨*proof*⟩

lemma *Ball-qbs-pred-countable*:
  assumes $\bigwedge i$::'*a* :: *countable*. $i \in I \Longrightarrow$ *qbs-pred* X (P i)
  shows *qbs-pred* X (λx. $\forall x \in I$. P i x)
  ⟨*proof*⟩

lemma *Ball-qbs-pred*:
  assumes *finite* I $\bigwedge i$. $i \in I \Longrightarrow$ *qbs-pred* X (P i)
  shows *qbs-pred* X (λx. $\forall x \in I$. P i x)
  ⟨*proof*⟩

lemma *Bex-qbs-pred-countable*:
  assumes $\bigwedge i$::'*a* :: *countable*. $i \in I \Longrightarrow$ *qbs-pred* X (P i)
  shows *qbs-pred* X (λx. $\exists x \in I$. P i x)
  ⟨*proof*⟩

lemma *Bex-qbs-pred*:
  assumes *finite* I $\bigwedge i$. $i \in I \Longrightarrow$ *qbs-pred* X (P i)
  shows *qbs-pred* X (λx. $\exists x \in I$. P i x)
  ⟨*proof*⟩

lemma *qbs-morphism-If-sub-qbs*:
  assumes [*qbs*]: *qbs-pred* X P
    and [*qbs*]: $f \in$ *sub-qbs* X $\{x \in$ *qbs-space* X. P $x\}$ $\rightarrow_Q$ Y $g \in$ *sub-qbs* X $\{x \in$ *qbs-space* X. $\neg$ P $x\}$ $\rightarrow_Q$ Y
   shows (λx. *if* P x *then* f x *else* g x) $\in X \rightarrow_Q Y$
⟨*proof*⟩

3.3.5 The Adjunction w.r.t. Ordering

lemma *l-mono*: *mono qbs-to-measure*
⟨*proof*⟩

lemma *r-mono*: *mono measure-to-qbs*
⟨*proof*⟩

lemma *rl-order-adjunction*:
  $X \leq$ *qbs-to-measure* $Y \longleftrightarrow$ *measure-to-qbs* $X \leq Y$
⟨*proof*⟩

end

4 The S-Finite Measure Monad

theory *Monad-QuasiBorel*
 imports
  Measure-QuasiBorel-Adjunction
  Kernels

begin

4.1 The s-Finite Measure Monad

- In the previous version:
  - A measure on $X = [X, \alpha, \mu]_\sim$
    - $\alpha \in M_X$
    - μ is an s-finite measure on $\mathbb{R}$
    - $(X, \alpha, \mu) \sim (X, \beta, \nu) \iff \alpha_* \mu = \beta_* \nu$
  - The s-finite measure monad: $\mathscr{M}(X) = \{p \mid p \text{ is a measure on } X\}$
- Current version: measures are not restricted to s-finite measures.
  - A measure on $X = [X, \alpha, \mu]_\sim$
    - $\alpha \in M_X$
    - μ is a measure on $\mathbb{R}$
    - $(X, \alpha, \mu) \sim (X, \beta, \nu) \iff \alpha_* \mu = \beta_* \nu$
  - The s-finite measure monad: $\mathscr{M}(X) = \{[X, \alpha, \mu]_\sim \mid \mu \text{ is s-finite}\}$
    The space of all measures: $\mathscr{M}_{\text{all}}(X) = \{p \mid p \text{ is a measure on } X\}$

4.1.1 Measures on Quasi-Borel spaces

locale *in-Mx* =
 fixes X :: *'a quasi-borel*
  and α :: *real* ⇒ *'a*
 assumes *in-Mx*[*simp*]:$\alpha \in$ *qbs-Mx X*
begin

lemma *α-measurable*[*measurable*]: $\alpha \in$ *borel* $\to_M$ *qbs-to-measure X*
 ⟨*proof*⟩

lemma *α-qbs-morphism*[*qbs*]: $\alpha \in$ *qbs-borel* $\to_Q$ *X*
 ⟨*proof*⟩

lemma *X-not-empty*: *qbs-space X* $\neq \{\}$
 ⟨*proof*⟩

lemma *inverse-UNIV*[*simp*]: α −‘ (*qbs-space X*) = *UNIV*
⟨*proof*⟩

end

locale *qbs-meas* = *in-Mx X* α
for *X* :: ′*a quasi-borel* **and** α **and** μ :: *real measure* +
assumes *mu-sets*[*measurable-cong*]: *sets* μ = *sets borel*
begin

lemma *mu-not-empty*: *space* $\mu \neq \{\}$
⟨*proof*⟩

end

lemma *qbs-meas-All*:
assumes $\alpha \in$ *qbs-Mx X measure-kernel M borel k x* $\in$ *space M*
shows *qbs-meas X* α (*k x*)
⟨*proof*⟩

locale *qbs-s-finite* = *qbs-meas* + *s-finite-measure* μ

lemma *qbs-s-finite-All*:
assumes $\alpha \in$ *qbs-Mx X s-finite-kernel M borel k x* $\in$ *space M*
shows *qbs-s-finite X* α (*k x*)
⟨*proof*⟩

locale *qbs-prob* = *in-Mx X* α + *real-distribution* μ
for *X* :: ′*a quasi-borel* **and** α μ
begin

lemma *qbs-meas*: *qbs-meas X* α μ
⟨*proof*⟩

lemma *qbs-s-finite*: *qbs-s-finite X* α μ
⟨*proof*⟩

sublocale *qbs-s-finite* ⟨*proof*⟩

end

locale *pair-qbs-meas′* = *pq1*: *qbs-meas X* α μ + *pq2*: *qbs-meas Y* β ν
for *X* :: ′*a quasi-borel* **and** α μ **and** *Y* :: ′*b quasi-borel* **and** β ν
begin

lemma *ab-measurable*[*measurable*]: *map-prod* α $\beta \in$ *borel* $\bigotimes_M$ *borel* $\rightarrow_M$ *qbs-to-measure* (*X* $\bigotimes_Q$ *Y*)
⟨*proof*⟩

end

locale *pair-qbs-meas* = *pq1*: *qbs-meas* X α μ + *pq2*: *qbs-meas* X β ν
for X :: *'a quasi-borel* **and** α μ β ν
begin

sublocale *pair-qbs-meas'* X α μ X β ν
⟨*proof*⟩

end

locale *pair-qbs-s-finites* = *pq1*: *qbs-s-finite* X α μ + *pq2*: *qbs-s-finite* Y β ν
for X :: *'a quasi-borel* **and** α μ **and** Y :: *'b quasi-borel* **and** β ν
begin

sublocale *pair-qbs-meas'* X α μ Y β ν
⟨*proof*⟩

end

locale *pair-qbs-s-finite* = *pq1*: *qbs-s-finite* X α μ + *pq2*: *qbs-s-finite* X β ν
for X :: *'a quasi-borel* **and** α μ **and** β ν
begin

sublocale *pair-qbs-s-finites* X α μ X β ν
⟨*proof*⟩

sublocale *pair-qbs-meas* X α μ β ν
⟨*proof*⟩

end

locale *pair-qbs-probs* = *pq1*: *qbs-prob* X α μ + *pq2*: *qbs-prob* Y β ν
for X :: *'a quasi-borel* **and** α μ **and** Y :: *'b quasi-borel* **and** β ν
begin

sublocale *pair-qbs-s-finites*
⟨*proof*⟩

end

locale *pair-qbs-prob* = *pq1*: *qbs-prob* X α μ + *pq2*: *qbs-prob* X β ν
for X :: *'a quasi-borel* **and** α μ **and** β ν
begin

sublocale *pair-qbs-s-finite* X α μ β ν
⟨*proof*⟩

sublocale *pair-qbs-probs* X α μ X β μ

⟨*proof*⟩

end

lemma(**in** *qbs-meas*) *qbs-probI*: *prob-space* $\mu \Longrightarrow$ *qbs-prob* $X\ \alpha\ \mu$
⟨*proof*⟩

type-synonym *'a qbs-measure-t* = *'a quasi-borel* ∗ (*real* ⇒ *'a*) ∗ *real measure*
definition *qbs-meas-eq* :: [*'a qbs-measure-t*, *'a qbs-measure-t*] ⇒ *bool* **where**
qbs-meas-eq p1 p2 ≡
(*let* (X, α, μ) = *p1*;
(Y, β, ν) = *p2 in*
qbs-meas $X\ \alpha\ \mu \wedge$ *qbs-meas* $Y\ \beta\ \nu \wedge X = Y \wedge$
distr μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* Y) β)

lemma *qbs-meas-eq-def2*:
qbs-meas-eq p1 p2 =
(*let* (X::*'a quasi-borel*, α, μ) = *p1*;
(Y, β, ν) = *p2 in*
qbs-meas $X\ \alpha\ \mu \wedge$ *qbs-meas* $Y\ \beta\ \nu \wedge X = Y \wedge$
($\forall f \in X \to_Q$ (*qbs-borel* :: *ennreal quasi-borel*). ($\int^+ x.\ f\ (\alpha\ x)\ \partial\mu$) = ($\int^+ x.\ f\ (\beta\ x)\ \partial\nu$)))
⟨*proof*⟩

lemma(**in** *qbs-meas*) *qbs-meas-eq-refl*[*simp*]: *qbs-meas-eq* (X,α,μ) (X,α,μ)
⟨*proof*⟩

lemma (**in** *pair-qbs-meas*)
shows *qbs-meas-eq-intro*:
distr μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* X) $\beta \Longrightarrow$ *qbs-meas-eq* (X,α,μ) (X,β,ν)
and *qbs-meas-eq-intro2*:
($\bigwedge f.\ f \in X \to_Q$ *qbs-borel* $\Longrightarrow$ ($\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu$) = ($\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu$)) $\Longrightarrow$ *qbs-meas-eq* (X,α,μ) (X,β,ν)
⟨*proof*⟩

lemma *qbs-meas-eq-dest*:
assumes *qbs-meas-eq* (X,α,μ) (Y,β,ν)
shows *qbs-meas* $X\ \alpha\ \mu$ *qbs-meas* $Y\ \beta\ \nu$ $Y = X$ *distr* μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* X) β
⟨*proof*⟩

lemma *qbs-meas-eq-dest2*:
assumes *qbs-meas-eq* (X,α,μ) (Y,β,ν)
shows *qbs-meas* $X\ \alpha\ \mu$ *qbs-meas* $Y\ \beta\ \nu$ $Y = X$ $\bigwedge f.\ f \in X \to_Q$ *qbs-borel* $\Longrightarrow$ ($\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu$) = ($\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu$)
⟨*proof*⟩

lemma *qbs-meas-eq-integral-eq*:

assumes *qbs-meas-eq* (X,α,μ) (Y,β,ν)
and [*measurable*]:$f \in X \to_Q$ (*qbs-borel* :: $'b$::{*banach*, *second-countable-topology*} *quasi-borel*)
shows $(\int x.\ f\ (\alpha\ x)\ \partial\mu) = (\int x.\ f\ (\beta\ x)\ \partial\nu)$
⟨*proof*⟩

lemma
shows *qbs-meas-eq-symp*: *symp qbs-meas-eq*
and *qbs-meas-eq-transp*: *transp qbs-meas-eq*
⟨*proof*⟩

quotient-type $'a$ *qbs-measure* = $'a$ *qbs-measure-t* / *partial*: *qbs-meas-eq*
morphisms *rep-qbs-measure qbs-measure*
⟨*proof*⟩

interpretation *qbs-measure* : *quot-type qbs-meas-eq Abs-qbs-measure Rep-qbs-measure*
⟨*proof*⟩

syntax
-qbs-measure :: $'a$ *quasi-borel* ⇒ (*real* ⇒ $'a$) ⇒ *real measure* ⇒ $'a$ *qbs-measure* (⟦-,/ -,/ -⟧$_{meas}$)
syntax-consts
-qbs-measure ⇌ *qbs-measure*
translations
⟦X, α, μ⟧$_{meas}$ ⇌ *CONST qbs-measure* (X, α, μ)

lemma *rep-qbs-measure'*: $\exists X\ \alpha\ \mu.\ p =$ ⟦X, α, μ⟧$_{meas}$ $\wedge$ *qbs-meas* $X\ \alpha\ \mu$
⟨*proof*⟩

lemma *rep-qbs-measure*:
obtains $X\ \alpha\ \mu$ **where** $p =$ ⟦X, α, μ⟧$_{meas}$ $\wedge$ *qbs-meas* $X\ \alpha\ \mu$
⟨*proof*⟩

definition *qbs-null-measure* :: $'a$ *quasi-borel* ⇒ $'a$ *qbs-measure* **where**
qbs-null-measure $X \equiv$ ⟦X, *SOME a. a* $\in$ *qbs-Mx X*, *null-measure borel*⟧$_{meas}$

lemma *qbs-null-measure-meas*: *qbs-space* $X \neq \{\} \Longrightarrow$ *qbs-meas X* (*SOME a. a* $\in$ *qbs-Mx X*) (*null-measure borel*)
and *qbs-null-measure-s-finite*: *qbs-space* $X \neq \{\} \Longrightarrow$ *qbs-s-finite X* (*SOME a. a* $\in$ *qbs-Mx X*) (*null-measure borel*)
⟨*proof*⟩

lemma *in-Rep-qbs-measure'*:
assumes *qbs-meas-eq* (X,α,μ) (X',α',μ')
shows $(X',\alpha',\mu') \in$ *Rep-qbs-measure* ⟦X, α, μ⟧$_{meas}$
⟨*proof*⟩

lemmas(**in** *qbs-meas*) *in-Rep-qbs-measure* = *in-Rep-qbs-measure'*[*OF qbs-meas-eq-refl*]

lemma(**in** *qbs-meas*) *in-Rep-qbs-measure-dest*:
assumes $(X',\alpha',\mu') \in$ *Rep-qbs-measure* $[\![X, \alpha, \mu]\!]_{meas}$
shows $X' = X$
qbs-meas X' α' μ'
qbs-meas-eq (X,α,μ) (X',α',μ')
⟨*proof*⟩

lemma(**in** *qbs-meas*) *in-Rep-qbs-measure-dest'*:
assumes $p \in$ *Rep-qbs-measure* $[\![X, \alpha, \mu]\!]_{meas}$
obtains α' μ' **where** $p = (X, \alpha', \mu')$ *qbs-meas* X α' μ' *qbs-meas-eq* (X,α,μ) (X,α',μ')
⟨*proof*⟩

lemma *qbs-meas-eqI'*: *qbs-meas-eq* (X,α,μ) $(Y,\beta,\nu) \Longrightarrow [\![X, \alpha, \mu]\!]_{meas} = [\![Y, \beta, \nu]\!]_{meas}$
⟨*proof*⟩

lemma(**in** *pair-qbs-meas*) *qbs-meas-eqI*:
distr μ (*qbs-to-measure* X) α = *distr* ν (*qbs-to-measure* X) $\beta \Longrightarrow [\![X, \alpha, \mu]\!]_{meas} = [\![X, \beta, \nu]\!]_{meas}$
⟨*proof*⟩

lemma(**in** *pair-qbs-meas*) *qbs-meas-eqI2*:
$(\bigwedge f.\ f \in X \rightarrow_Q$ *qbs-borel* $\Longrightarrow (\int^+ x.\ f\ (\alpha\ x)\ \partial\ \mu) = (\int^+ x.\ f\ (\beta\ x)\ \partial\ \nu)) \Longrightarrow [\![X, \alpha, \mu]\!]_{meas} = [\![X, \beta, \nu]\!]_{meas}$
⟨*proof*⟩

lemma(**in** *pair-qbs-s-finite*) *qbs-s-finite-measure-eq-inverse*:
assumes $[\![X, \alpha, \mu]\!]_{meas} = [\![X, \beta, \nu]\!]_{meas}$
shows *qbs-meas-eq* (X,α,μ) (X,β,ν)
⟨*proof*⟩

lift-definition *qbs-space-of* :: *'a qbs-measure* ⇒ *'a quasi-borel*
is *fst* ⟨*proof*⟩

lemma (**in** *qbs-meas*) *qbs-space-of*[*simp*]: *qbs-space-of* $[\![X, \alpha, \mu]\!]_{meas} = X$
⟨*proof*⟩

lemma *qbs-space-of-non-empty*: *qbs-space* (*qbs-space-of* p) $\neq \{\}$
⟨*proof*⟩

4.1.2 The Space of All Measures

definition *all-meas-qbs* :: *'a quasi-borel* ⇒ *'a qbs-measure quasi-borel* **where**
all-meas-qbs $X \equiv$ *Abs-quasi-borel* ($\{s.$ *qbs-space-of* $s = X\}$, $\{\lambda r.\ [\![X, \alpha, k\ r]\!]_{meas}$ $|\alpha\ k.\ \alpha \in$ *qbs-Mx* $X \wedge$ *measure-kernel borel borel* $k\}$)

lemma
shows *all-meas-qbs-space*: *qbs-space* (*all-meas-qbs* X) = $\{s.$ *qbs-space-of* $s = X\}$ (**is** *?g1*)
and *all-meas-qbs-Mx*: *qbs-Mx* (*all-meas-qbs* X) = $\{\lambda r.$ $[\![X,\ \alpha,\ k\ r]\!]_{meas}$ $|\alpha\ k.\ \alpha$ $\in$ *qbs-Mx* $X \wedge$ *measure-kernel borel borel* $k\}$ (**is** *?g2*)
⟨*proof*⟩

lemma *all-meas-qbs-empty-iff*: *qbs-space* $X = \{\} \longleftrightarrow$ *qbs-space* (*all-meas-qbs* X) $= \{\}$
⟨*proof*⟩

lemma(**in** *qbs-meas*) *in-space-all-meas*[*qbs*]: $[\![X,\ \alpha,\ \mu]\!]_{meas} \in$ *qbs-space* (*all-meas-qbs* X)
⟨*proof*⟩

lemma *rep-qbs-space-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* X)
obtains α μ **where** $s = [\![X,\ \alpha,\ \mu]\!]_{meas}$ *qbs-meas* X α μ
⟨*proof*⟩

lemma *qbs-space-of-in-all-meas*: $s \in$ *qbs-space* (*all-meas-qbs* X) $\Longrightarrow$ *qbs-space-of* s $= X$
⟨*proof*⟩

lemma *in-qbs-space-of-all-meas*: $s \in$ *qbs-space* (*all-meas-qbs* (*qbs-space-of* s))
⟨*proof*⟩

4.1.3 *l*

lift-definition *qbs-l* :: *'a qbs-measure* $\Rightarrow$ *'a measure*
is $\lambda(X,\alpha,\mu)$. *distr* μ (*qbs-to-measure* X) α
⟨*proof*⟩

lemma(**in** *qbs-meas*) *qbs-l*: *qbs-l* $[\![X,\ \alpha,\ \mu]\!]_{meas}$ = *distr* μ (*qbs-to-measure* X) α
⟨*proof*⟩

lemma *space-qbs-l*: *qbs-space* (*qbs-space-of* s) = *space* (*qbs-l* s)
⟨*proof*⟩

lemma *space-qbs-l-ne*: *space* (*qbs-l* s) $\neq \{\}$
⟨*proof*⟩

lemma *qbs-l-sets*: *sets* (*qbs-to-measure* (*qbs-space-of* s)) = *sets* (*qbs-l* s)
⟨*proof*⟩

lemma *qbs-null-measure-in-all-meas*: *qbs-space* $X \neq \{\} \Longrightarrow$ *qbs-null-measure* $X \in$ *qbs-space* (*all-meas-qbs* X)
⟨*proof*⟩

lemma *qbs-null-measure-null-measure*:*qbs-space* $X \neq \{\} \Longrightarrow$ *qbs-l* (*qbs-null-measure* X) = *null-measure* (*qbs-to-measure* X)
⟨*proof*⟩

lemma *space-qbs-l-in-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* X)
shows *space* (*qbs-l* s) = *qbs-space* X
⟨*proof*⟩

lemma *sets-qbs-l-all-meaures*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* X)
shows *sets* (*qbs-l* s) = *sets* (*qbs-to-measure* X)
⟨*proof*⟩

lemma *measurable-qbs-l-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* X)
shows *qbs-l* $s \rightarrow_M M = X \rightarrow_Q$ *measure-to-qbs* M
⟨*proof*⟩

lemma *measurable-qbs-l-all-meas′*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* X)
shows *qbs-l* $s \rightarrow_M M$ = *qbs-to-measure* $X \rightarrow_M M$
⟨*proof*⟩

lemma *rep-all-meas-qbs-Mx*:
assumes $\gamma \in$ *qbs-Mx* (*all-meas-qbs* X)
obtains α k **where** $\gamma = (\lambda r.\ [\![X,\ \alpha,\ k\ r]\!]_{meas})$ $\alpha \in$ *qbs-Mx* X *measure-kernel borel borel* k $\bigwedge r.$ *qbs-meas* X α (k r)
⟨*proof*⟩

lemma *qbs-l-measure-kernel-all-meas*:
measure-kernel (*qbs-to-measure* (*all-meas-qbs* X)) (*qbs-to-measure* X) *qbs-l*
⟨*proof*⟩

lemma *qbs-l-inj-all-meas*: *inj-on qbs-l* (*qbs-space* (*all-meas-qbs* X))
⟨*proof*⟩

lemma *qbs-l-morphism-all-meas*:
assumes [*measurable*]:$A \in$ *sets* (*qbs-to-measure* X)
shows ($\lambda s.$ *qbs-l* s A) $\in$ *all-meas-qbs* $X \rightarrow_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-l-finite-pred-all-meas*: *qbs-pred* (*all-meas-qbs* X) ($\lambda s.$ *finite-measure* (*qbs-l* s))
⟨*proof*⟩

lemma *qbs-l-subprob-pred-all-meas*: *qbs-pred* (*all-meas-qbs* X) ($\lambda s.$ *subprob-space* (*qbs-l* s))
⟨*proof*⟩

lemma *qbs-l-prob-pred-all-meas*: *qbs-pred* (*all-meas-qbs X*) (λs. *prob-space* (*qbs-l s*))
⟨*proof*⟩

4.1.4 Return

definition *return-qbs* :: *'a quasi-borel* ⇒ *'a* ⇒ *'a qbs-measure* **where**
return-qbs X x ≡ $[\![X, \lambda r.\ x,\ SOME\ \mu.\ real\text{-}distribution\ \mu]\!]_{meas}$

lemma(**in** *real-distribution*)
assumes $x \in$ *qbs-space X*
shows *return-qbs*:*return-qbs X x* = $[\![X, \lambda r.\ x, M]\!]_{meas}$
and *return-qbs-meas*:*qbs-meas X* ($\lambda r.\ x$) *M*
and *return-qbs-prob*:*qbs-prob X* ($\lambda r.\ x$) *M*
and *return-qbs-s-finite*:*qbs-s-finite X* ($\lambda r.\ x$) *M*
⟨*proof*⟩

lemma *return-qbs-comp*:
assumes $\alpha \in$ *qbs-Mx X*
shows (*return-qbs X* ∘ α) = (λr. $[\![X, \alpha,\ return\ borel\ r]\!]_{meas}$)
⟨*proof*⟩

corollary *return-qbs-morphism-all-meas*: *return-qbs X* ∈ $X \to_Q$ *all-meas-qbs X*
⟨*proof*⟩

4.1.5 Bind

definition *bind-qbs* :: [*'a qbs-measure*, *'a* ⇒ *'b qbs-measure*] ⇒ *'b qbs-measure* **where**
bind-qbs s f ≡ (*let* (X, α, μ) = *rep-qbs-measure s*;
Y = *qbs-space-of* (f (α *undefined*));
(β, k) = (*SOME* (β, k). $f \circ \alpha$ = (λr. $[\![Y, \beta, k\ r]\!]_{meas}$) ∧ β ∈ *qbs-Mx Y* ∧ *measure-kernel borel borel k*) *in*
$[\![Y, \beta, \mu \gg\!=_k k]\!]_{meas}$)

adhoc-overloading *Monad-Syntax.bind* ⇌ *bind-qbs*

lemma(**in** *qbs-meas*)
assumes $s = [\![X, \alpha, \mu]\!]_{meas}$
$f \in X \to_Q$ *all-meas-qbs Y*
$\beta \in$ *qbs-Mx Y*
measure-kernel borel borel k
and ($f \circ \alpha$) = (λr. $[\![Y, \beta, k\ r]\!]_{meas}$)
shows *bind-qbs-meas*:*qbs-meas Y* β ($\mu \gg\!=_k k$)
and *bind-qbs-all-meas*: $s \gg\!= f = [\![Y, \beta, \mu \gg\!=_k k]\!]_{meas}$
⟨*proof*⟩

lemma *bind-qbs-morphism-all-meas'*:

assumes $f \in X \to_Q$ *all-meas-qbs* Y
shows $(\lambda x.\ x \gg= f) \in$ *all-meas-qbs* $X \to_Q$ *all-meas-qbs* Y
⟨*proof*⟩

lemma *bind-qbs-return-all-meas'*:
assumes $x \in$ *qbs-space* (*all-meas-qbs* X)
shows $x \gg=$ *return-qbs* $X = x$
⟨*proof*⟩

lemma *bind-qbs-return-all-meas*:
assumes $f \in X \to_Q$ *all-meas-qbs* Y
and $x \in$ *qbs-space* X
shows *return-qbs* $X\ x \gg= f = f\ x$
⟨*proof*⟩

Associativity seems not to hold for *all-meas-qbs*.

lemma *bind-qbs-cong-all-meas*:
assumes [*qbs*]:$s \in$ *qbs-space* (*all-meas-qbs* X)
$\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x = g\ x$
and [*qbs*]:$f \in X \to_Q$ *all-meas-qbs* Y
shows $s \gg= f = s \gg= g$
⟨*proof*⟩

4.1.6 The Functorial Action

definition *distr-qbs* :: ['*a quasi-borel*, '*b quasi-borel*,'*a* ⇒ '*b*,'*a qbs-measure*] ⇒ '*b qbs-measure* **where**
distr-qbs - $Y\ f\ sx \equiv sx \gg=$ *return-qbs* $Y \circ f$

lemma *distr-qbs-morphism-all-meas'*:
assumes $f \in X \to_Q Y$
shows *distr-qbs* $X\ Y\ f \in$ *all-meas-qbs* $X \to_Q$ *all-meas-qbs* Y
⟨*proof*⟩

lemma(**in** *qbs-meas*)
assumes $s = [\![X, \alpha, \mu]\!]_{meas}$
and $f \in X \to_Q Y$
shows *distr-qbs-meas*:*qbs-meas* $Y\ (f \circ \alpha)\ \mu$
and *distr-qbs*: *distr-qbs* $X\ Y\ f\ s = [\![Y, f \circ \alpha, \mu]\!]_{meas}$
⟨*proof*⟩

lemma(**in** *qbs-s-finite*) *distr-qbs-s-finite*:
assumes [*qbs*]:$f \in X \to_Q Y$
shows *qbs-s-finite* $Y\ (f \circ \alpha)\ \mu$
⟨*proof*⟩

lemma(**in** *qbs-prob*) *distr-qbs-prob*:
assumes [*qbs*]: $f \in X \to_Q Y$
shows *qbs-prob* $Y\ (f \circ \alpha)\ \mu$

⟨*proof*⟩

lemma *distr-qbs-id-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
shows *distr-qbs X X id s* = *s*
⟨*proof*⟩

lemma *distr-qbs-comp-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
$f \in X \to_Q Y$
and $g \in Y \to_Q Z$
shows ((*distr-qbs Y Z g*) ∘ (*distr-qbs X Y f*)) *s* = *distr-qbs X Z* (*g* ∘ *f*) *s*
⟨*proof*⟩

4.1.7 Join

definition *join-qbs* :: *'a qbs-measure qbs-measure* ⇒ *'a qbs-measure* **where**
join-qbs ≡ (λ*sst*. *sst* ≫= *id*)

lemma *join-qbs-morphism-all-meas*: *join-qbs* ∈ *all-meas-qbs* (*all-meas-qbs X*) $\to_Q$ *all-meas-qbs X*
⟨*proof*⟩

lemma
assumes *qbs-meas* (*all-meas-qbs X*) β μ
ssx = $\llbracket$*all-meas-qbs X*, β, μ$\rrbracket_{meas}$
α ∈ *qbs-Mx X*
measure-kernel borel borel k
and β =(λ*r*. $\llbracket$*X*, α, *k r*$\rrbracket_{meas}$)
shows *join-qbs-meas*: *qbs-meas X* α (μ $\gg\!=_k$ *k*)
and *join-qbs-all-meas*: *join-qbs ssx* = $\llbracket$*X*, α, μ $\gg\!=_k$ *k*$\rrbracket_{meas}$
⟨*proof*⟩

4.1.8 Strength

definition *strength-qbs* :: [*'a quasi-borel*,*'b quasi-borel*,*'a* × *'b qbs-measure*] ⇒ (*'a* × *'b*) *qbs-measure* **where**
strength-qbs W X = (λ(*w*,*sx*). *let* (-,α,μ) = *rep-qbs-measure sx*
in $\llbracket$*W* $\bigotimes_Q$ *X*, λ*r*. (*w*, α *r*), μ$\rrbracket_{meas}$)

lemma(**in** *qbs-meas*)
assumes [*qbs*]:*w* ∈ *qbs-space W*
and *sx* = $\llbracket$*X*, α, μ$\rrbracket_{meas}$
shows *strength-qbs-meas*: *qbs-meas* (*W* $\bigotimes_Q$ *X*) (λ*r*. (*w*,α *r*)) μ
and *strength-qbs*: *strength-qbs W X* (*w*,*sx*) = $\llbracket$*W* $\bigotimes_Q$ *X*, λ*r*. (*w*,α *r*), μ$\rrbracket_{meas}$
⟨*proof*⟩

lemma(**in** *qbs-s-finite*) *strength-qbs-s-finite*: *w* ∈ *qbs-space W* ⟹ *qbs-s-finite* (*W* $\bigotimes_Q$ *X*) (λ*r*. (*w*,α *r*)) μ
⟨*proof*⟩

lemma(**in** *qbs-prob*) *strength-qbs-prob*: $w \in$ *qbs-space* $W \Longrightarrow$ *qbs-prob* ($W \bigotimes_Q X$) ($\lambda r.$ $(w,\alpha\ r)$) μ
⟨*proof*⟩

lemma *strength-qbs-natural-all-meas*:
assumes [*qbs*]:$f \in X \to_Q X'$ $g \in Y \to_Q Y'$ $x \in$ *qbs-space* X *sy* $\in$ *qbs-space* (*all-meas-qbs* Y)
shows (*distr-qbs* ($X \bigotimes_Q Y$) ($X' \bigotimes_Q Y'$) (*map-prod* f g) $\circ$ *strength-qbs* X Y) (x,*sy*) = (*strength-qbs* X' Y' $\circ$ *map-prod* f (*distr-qbs* Y Y' g)) (x,*sy*)
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *strength-qbs-law1-all-meas*:
assumes $x \in$ *qbs-space* (*unit-quasi-borel* $\bigotimes_Q$ *all-meas-qbs* X)
shows *snd* x = (*distr-qbs* (*unit-quasi-borel* $\bigotimes_Q X$) X *snd* $\circ$ *strength-qbs unit-quasi-borel* X) x
⟨*proof*⟩

lemma *strength-qbs-law2-all-meas*:
assumes $x \in$ *qbs-space* (($X \bigotimes_Q Y$) $\bigotimes_Q$ *all-meas-qbs* Z)
shows (*strength-qbs* X ($Y \bigotimes_Q Z$) $\circ$ (*map-prod id* (*strength-qbs* Y Z)) $\circ$ ($\lambda((x,y),z).$ $(x,(y,z))$)) x =
(*distr-qbs* (($X \bigotimes_Q Y$) $\bigotimes_Q Z$) ($X \bigotimes_Q$ ($Y \bigotimes_Q Z$)) ($\lambda((x,y),z).$ $(x,(y,z))$) $\circ$ *strength-qbs* ($X \bigotimes_Q Y$) Z) x
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

4.1.9 The s-Finite Measure Monad

definition *monadM-qbs* :: *'a quasi-borel* $\Rightarrow$ *'a qbs-measure quasi-borel* **where**
monadM-qbs $X \equiv$ *Abs-quasi-borel* ($\{[\![X, \alpha, \mu]\!]_{meas} \mid \alpha\ \mu.$ *qbs-s-finite* $X\ \alpha\ \mu\}$, $\{\lambda r.$ $[\![X, \alpha, k\ r]\!]_{meas} \mid \alpha\ k.$ $\alpha \in$ *qbs-Mx* $X \wedge$ *s-finite-kernel borel borel* $k\}$)

lemma
shows *monadM-qbs-space*: *qbs-space* (*monadM-qbs* X) = $\{[\![X, \alpha, \mu]\!]_{meas} \mid \alpha\ \mu.$ *qbs-s-finite* $X\ \alpha\ \mu\}$
and *monadM-qbs-Mx*: *qbs-Mx* (*monadM-qbs* X) = $\{\lambda r.$ $[\![X, \alpha, k\ r]\!]_{meas} \mid \alpha\ k.$ $\alpha \in$ *qbs-Mx* $X \wedge$ *s-finite-kernel borel borel* $k\}$
⟨*proof*⟩

lemma *monadM-all-meas-space'*: *qbs-space* (*monadM-qbs* X) $\subseteq$ *qbs-space* (*all-meas-qbs* X)
and *monadM-all-meas-space*: $\bigwedge p.$ $p \in$ *qbs-space* (*monadM-qbs* X) $\Longrightarrow p \in$ *qbs-space* (*all-meas-qbs* X)
and *monadM-all-meas-Mx*: *qbs-Mx* (*monadM-qbs* X) $\subseteq$ *qbs-Mx* (*all-meas-qbs* X)
⟨*proof*⟩

lemma

shows *qbs-morphism-monadAD*: $f \in X \to_Q$ *monadM-qbs* $Y \Longrightarrow f \in X \to_Q$ *all-meas-qbs* Y
and *qbs-morphism-monadAD'*: $g \in$ *all-meas-qbs* $X \to_Q Y \Longrightarrow g \in$ *monadM-qbs* $X \to_Q Y$
⟨*proof*⟩

lemma *monadM-qbs-empty-iff*: *qbs-space* $X = \{\} \longleftrightarrow$ *qbs-space* (*monadM-qbs* X) $= \{\}$
⟨*proof*⟩

lemma(**in** *qbs-s-finite*) *in-space-monadM*[*qbs*]: $[\![X, \alpha, \mu]\!]_{meas} \in$ *qbs-space* (*monadM-qbs* X)
⟨*proof*⟩

lemma *rep-qbs-space-monadM*:
assumes $s \in$ *qbs-space* (*monadM-qbs* X)
obtains α μ **where** $s = [\![X, \alpha, \mu]\!]_{meas}$ *qbs-s-finite* X α μ
⟨*proof*⟩

lemma *rep-qbs-space-monadM-sigma-finite*:
assumes $s \in$ *qbs-space* (*monadM-qbs* X)
obtains α μ **where** $s = [\![X, \alpha, \mu]\!]_{meas}$ *qbs-s-finite* X α μ *sigma-finite-measure* μ
⟨*proof*⟩

lemma *qbs-space-of-in*: $s \in$ *qbs-space* (*monadM-qbs* X) $\Longrightarrow$ *qbs-space-of* $s = X$
⟨*proof*⟩

lemma *qbs-l-s-finite*:
assumes $p \in$ *qbs-space* (*monadM-qbs* X)
shows *s-finite-measure* (*qbs-l* p)
⟨*proof*⟩

lemma *qbs-null-measure-in-Mx*: *qbs-space* $X \neq \{\} \Longrightarrow$ *qbs-null-measure* $X \in$ *qbs-space* (*monadM-qbs* X)
⟨*proof*⟩

lemma *space-qbs-l-in*:
assumes $s \in$ *qbs-space* (*monadM-qbs* X)
shows *space* (*qbs-l* s) = *qbs-space* X
⟨*proof*⟩

lemma *sets-qbs-l*:
assumes $s \in$ *qbs-space* (*monadM-qbs* X)
shows *sets* (*qbs-l* s) = *sets* (*qbs-to-measure* X)
⟨*proof*⟩

lemma *measurable-qbs-l*:
assumes $s \in$ *qbs-space* (*monadM-qbs* X)

shows *qbs-l s* $\to_M$ *M* = *X* $\to_Q$ *measure-to-qbs M*
⟨*proof*⟩

lemma *measurable-qbs-l'*:
assumes *s* ∈ *qbs-space* (*monadM-qbs X*)
shows *qbs-l s* $\to_M$ *M* = *qbs-to-measure X* $\to_M$ *M*
⟨*proof*⟩

lemma *rep-qbs-Mx-monadM*:
assumes γ ∈ *qbs-Mx* (*monadM-qbs X*)
obtains α *k* **where** γ = (λ*r*. ⟦*X*, α, *k r*⟧$_{meas}$) α ∈ *qbs-Mx X s-finite-kernel borel borel k* ⋀*r*. *qbs-s-finite X* α (*k r*)
⟨*proof*⟩

lemma *qbs-l-measurable*[*measurable*]:*qbs-l* ∈ *qbs-to-measure* (*monadM-qbs X*) $\to_M$ *s-finite-measure-algebra* (*qbs-to-measure X*)
⟨*proof*⟩

lemma *qbs-l-measure-kernel*: *measure-kernel* (*qbs-to-measure* (*monadM-qbs X*)) (*qbs-to-measure X*) *qbs-l*
⟨*proof*⟩

lemmas *qbs-l-inj* = *inj-on-subset*[*OF qbs-l-inj-all-meas monadM-all-meas-space'*]

lemmas *qbs-l-morphism* = *qbs-morphism-monadAD'*[*OF qbs-l-morphism-all-meas*]

lemmas *qbs-l-finite-pred* = *qbs-morphism-monadAD'*[*OF qbs-l-finite-pred-all-meas*]

lemmas *qbs-l-subprob-pred* = *qbs-morphism-monadAD'*[*OF qbs-l-subprob-pred-all-meas*]

lemmas *qbs-l-prob-pred* = *qbs-morphism-monadAD'*[*OF qbs-l-prob-pred-all-meas*]

lemma *return-qbs-morphism*[*qbs*]: *return-qbs X* ∈ *X* $\to_Q$ *monadM-qbs X*
⟨*proof*⟩

lemma(**in** *qbs-s-finite*)
assumes *s* = ⟦*X*, α, μ⟧$_{meas}$
f ∈ *X* $\to_Q$ *monadM-qbs Y*
β ∈ *qbs-Mx Y*
s-finite-kernel borel borel k
and (*f* ∘ α) = (λ*r*. ⟦*Y*, β, *k r*⟧$_{meas}$)
shows *bind-qbs-s-finite*:*qbs-s-finite Y* β (μ $\gg\!=_k$ *k*)
and *bind-qbs*: *s* ≫= *f* = ⟦*Y*, β, μ $\gg\!=_k$ *k*⟧$_{meas}$
⟨*proof*⟩

lemma *bind-qbs-morphism'*:
assumes *f* ∈ *X* $\to_Q$ *monadM-qbs Y*
shows (λ*x*. *x* ≫= *f*) ∈ *monadM-qbs X* $\to_Q$ *monadM-qbs Y*
⟨*proof*⟩

lemmas *bind-qbs-return′* = *bind-qbs-return-all-meas′*[*OF monadM-all-meas-space*]

lemmas *bind-qbs-return* = *bind-qbs-return-all-meas*[*OF qbs-morphism-monadAD*]

lemma *bind-qbs-assoc*:
  assumes $s \in$ *qbs-space* (*monadM-qbs X*)
    $f \in X \to_Q$ *monadM-qbs Y*
   and $g \in Y \to_Q$ *monadM-qbs Z*
  shows $s \gg\!= (\lambda x.\ f\ x \gg\!= g) = (s \gg\!= f) \gg\!= g$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *bind-qbs-cong*:
  assumes [*qbs*]:$s \in$ *qbs-space* (*monadM-qbs X*)
    $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x = g\ x$
   and [*qbs*]:$f \in X \to_Q$ *monadM-qbs Y*
  shows $s \gg\!= f = s \gg\!= g$
⟨*proof*⟩

lemma *distr-qbs-morphism′*:
  assumes $f \in X \to_Q Y$
  shows *distr-qbs X Y f* $\in$ *monadM-qbs X* $\to_Q$ *monadM-qbs Y*
  ⟨*proof*⟩

We show that M is a functor i.e. M preserve identity and composition.

lemma *distr-qbs-id*:
  assumes $s \in$ *qbs-space* (*monadM-qbs X*)
  shows *distr-qbs X X id s* = *s*
  ⟨*proof*⟩

lemma *distr-qbs-comp*:
  assumes [*qbs*]:$s \in$ *qbs-space* (*monadM-qbs X*) $f \in X \to_Q Y$ $g \in Y \to_Q Z$
  shows ((*distr-qbs Y Z g*) $\circ$ (*distr-qbs X Y f*)) *s* = *distr-qbs X Z* ($g \circ f$) *s*
  ⟨*proof*⟩

lemma *join-qbs-morphism*[*qbs*]: *join-qbs* $\in$ *monadM-qbs* (*monadM-qbs X*) $\to_Q$ *monadM-qbs X*
  ⟨*proof*⟩

lemma
  assumes *qbs-s-finite* (*monadM-qbs X*) β μ
    ssx = $[\![$*monadM-qbs X*, β, $\mu]\!]_{meas}$
    $\alpha \in$ *qbs-Mx X*
    s-finite-kernel borel borel k
   and $\beta = (\lambda r.\ [\![X, \alpha, k\ r]\!]_{meas})$
  shows *join-qbs-s-finite*: *qbs-s-finite X* α ($\mu \gg\!=_k k$)
   and *join-qbs*: *join-qbs ssx* = $[\![X, \alpha, \mu \gg\!=_k k]\!]_{meas}$
  ⟨*proof*⟩

lemma *strength-qbs-natural*:
assumes [*qbs*]:$f \in X \to_Q X'$ $g \in Y \to_Q Y'$ $x \in$ *qbs-space* X $sy \in$ *qbs-space* (*monadM-qbs* Y)
shows (*distr-qbs* ($X \bigotimes_Q Y$) ($X' \bigotimes_Q Y'$) (*map-prod* f g) $\circ$ *strength-qbs* X Y) (x,sy) = (*strength-qbs* X' Y' $\circ$ *map-prod* f (*distr-qbs* Y Y' g)) (x,sy)
⟨*proof*⟩

context
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma *rr-from-real-to-real-id*[*simp*]: *rr.from-real* (*rr.to-real* x) = x *rr.from-real* $\circ$ *rr.to-real* = *id*
⟨*proof*⟩

lemma
assumes $\alpha \in$ *qbs-Mx* X
$\beta \in$ *qbs-Mx* (*monadM-qbs* Y)
$\gamma \in$ *qbs-Mx* Y
s-finite-kernel borel borel k
and $\beta = (\lambda r.\ [\![Y, \gamma, k\ r]\!]_{meas})$
shows *strength-qbs-ab-r-s-finite*: *qbs-s-finite* ($X \bigotimes_Q Y$) (*map-prod* α γ $\circ$ *rr.from-real*) (*distr* (*return borel* r $\bigotimes_M$ k r) *borel rr.to-real*)
and *strength-qbs-ab-r*: *strength-qbs* X Y (α r, β r) = $[\![X \bigotimes_Q Y$, *map-prod* α γ $\circ$ *rr.from-real*, *distr* (*return borel* r $\bigotimes_M$ k r) *borel rr.to-real*$]\!]_{meas}$ (**is** *?goal2*)
⟨*proof*⟩

lemma *strength-qbs-morphism*[*qbs*]: *strength-qbs* X Y $\in X \bigotimes_Q$ *monadM-qbs* Y $\to_Q$ *monadM-qbs* ($X \bigotimes_Q Y$)
⟨*proof*⟩

lemma *bind-qbs-morphism*[*qbs*]: ($\gg=$) $\in$ *monadM-qbs* $X \to_Q$ ($X \Rightarrow_Q$ *monadM-qbs* Y) $\Rightarrow_Q$ *monadM-qbs* Y
⟨*proof*⟩

lemma *strength-qbs-law1*:
$x \in$ *qbs-space* (*unit-quasi-borel* $\bigotimes_Q$ *monadM-qbs* X)
$\Longrightarrow$ *snd* x = (*distr-qbs* (*unit-quasi-borel* $\bigotimes_Q X$) X *snd* $\circ$ *strength-qbs unit-quasi-borel* X) x
⟨*proof*⟩

lemma *strength-qbs-law2*:
$x \in$ *qbs-space* (($X \bigotimes_Q Y$) $\bigotimes_Q$ *monadM-qbs* Z)
$\Longrightarrow$ (*strength-qbs* X ($Y \bigotimes_Q Z$) $\circ$ (*map-prod id* (*strength-qbs* Y Z)) $\circ$ ($\lambda((x,y),z).\ (x,(y,z))$)) x =
(*distr-qbs* (($X \bigotimes_Q Y$) $\bigotimes_Q Z$) ($X \bigotimes_Q$ ($Y \bigotimes_Q Z$)) ($\lambda((x,y),z).\ (x,(y,z))$) $\circ$ *strength-qbs* ($X \bigotimes_Q Y$) Z) x

⟨*proof*⟩

lemma *strength-qbs-law3*:
assumes $x \in$ *qbs-space* ($X \bigotimes_Q Y$)
shows *return-qbs* ($X \bigotimes_Q Y$) x = (*strength-qbs X Y* ∘ (*map-prod id* (*return-qbs* Y))) x
⟨*proof*⟩

lemma *strength-qbs-law4*:
assumes $x \in$ *qbs-space* ($X \bigotimes_Q$ *monadM-qbs* (*monadM-qbs* Y))
shows (*strength-qbs X Y* ∘ *map-prod id join-qbs*) x = (*join-qbs* ∘ *distr-qbs* ($X \bigotimes_Q$ *monadM-qbs* Y) (*monadM-qbs* ($X \bigotimes_Q Y$)) (*strength-qbs X Y*) ∘ *strength-qbs X* (*monadM-qbs* Y)) x
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *distr-qbs-morphism*[*qbs*]: *distr-qbs X Y* ∈ ($X \Rightarrow_Q Y$) $\rightarrow_Q$ (*monadM-qbs X* $\Rightarrow_Q$ *monadM-qbs Y*)
⟨*proof*⟩

lemma
assumes $\alpha \in$ *qbs-Mx X* $\beta \in$ *qbs-Mx Y*
shows *return-qbs-pair-Mx*: *return-qbs* ($X \bigotimes_Q Y$) (α *r*, β *k*) = ⟦$X \bigotimes_Q Y$,*map-prod* α β ∘ *rr.from-real*, *distr* (*return borel r* $\bigotimes_M$ *return borel k*) *borel rr.to-real*⟧$_{meas}$
and *return-qbs-pair-Mx-prob*: *qbs-prob* ($X \bigotimes_Q Y$) (*map-prod* α β ∘ *rr.from-real*) (*distr* (*return borel r* $\bigotimes_M$ *return borel k*) *borel rr.to-real*)
⟨*proof*⟩

lemma *bind-bind-return-distr*:
assumes *s-finite-measure* μ
and *s-finite-measure* ν
and [*measurable-cong*]: *sets* μ = *sets borel sets* ν = *sets borel*
shows $\mu \gg\!=_k$ (λr. $\nu \gg\!=_k$ (λl. *distr* (*return borel r* $\bigotimes_M$ *return borel l*) *borel rr.to-real*))
= *distr* ($\mu \bigotimes_M \nu$) *borel rr.to-real*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

end

context
begin
interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma *from-real-rr-qbs-morphism*[*qbs*]: *rr.from-real* ∈ *qbs-borel* $\rightarrow_Q$ *qbs-borel* $\bigotimes_Q$ *qbs-borel*
⟨*proof*⟩

end

context *pair-qbs-s-finites*
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

sublocale *qbs-s-finite* $X \bigotimes_Q Y$ *map-prod* α β ∘ *rr.from-real distr* ($\mu \bigotimes_M \nu$) *borel rr.to-real*
⟨*proof*⟩

lemma *qbs-bind-bind-return-qp*:
$[\![Y,\beta,\nu]\!]_{meas} \ggg= (\lambda y.\ [\![X,\alpha,\mu]\!]_{meas} \ggg= (\lambda x.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y))) = [\![X \bigotimes_Q Y,\ map\text{-}prod\ \alpha\ \beta \circ rr.from\text{-}real,\ distr\ (\mu \bigotimes_M \nu)\ borel\ rr.to\text{-}real]\!]_{meas}$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-bind-bind-return-pq*:
$[\![X,\alpha,\mu]\!]_{meas} \ggg= (\lambda x.\ [\![Y,\beta,\nu]\!]_{meas} \ggg= (\lambda y.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y))) = [\![X \bigotimes_Q Y,\ map\text{-}prod\ \alpha\ \beta \circ rr.from\text{-}real,\ distr\ (\mu \bigotimes_M \nu)\ borel\ rr.to\text{-}real]\!]_{meas}$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

end

lemma *bind-qbs-return-rotate*:
assumes $p \in$ *qbs-space* (*monadM-qbs X*)
and $q \in$ *qbs-space* (*monadM-qbs Y*)
shows $q \ggg= (\lambda y.\ p \ggg= (\lambda x.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y))) = p \ggg= (\lambda x.\ q \ggg= (\lambda y.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y)))$
⟨*proof*⟩

lemma *qbs-bind-bind-return1*:
assumes [*qbs*]: $f \in X \bigotimes_Q Y \to_Q$ *monadM-qbs Z*
$p \in$ *qbs-space* (*monadM-qbs X*)
$q \in$ *qbs-space* (*monadM-qbs Y*)
shows $q \ggg= (\lambda y.\ p \ggg= (\lambda x.\ f\ (x,y))) = (q \ggg= (\lambda y.\ p \ggg= (\lambda x.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y)))) \ggg= f$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-bind-bind-return2*:
assumes [*qbs*]:$f \in X \bigotimes_Q Y \to_Q$ *monadM-qbs Z*
$p \in$ *qbs-space* (*monadM-qbs X*) $q \in$ *qbs-space* (*monadM-qbs Y*)
shows $p \ggg= (\lambda x.\ q \ggg= (\lambda y.\ f\ (x,y))) = (p \ggg= (\lambda x.\ q \ggg= (\lambda y.\ return\text{-}qbs\ (X \bigotimes_Q Y)\ (x,y)))) \ggg= f$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

corollary *bind-qbs-rotate*:
assumes $f \in X \bigotimes_Q Y \to_Q$ *monadM-qbs Z*
$p \in$ *qbs-space* (*monadM-qbs X*)
and $q \in$ *qbs-space* (*monadM-qbs Y*)
shows $q \ggg= (\lambda y.\ p \ggg= (\lambda x.\ f\ (x,y))) = p \ggg= (\lambda x.\ q \ggg= (\lambda y.\ f\ (x,y)))$
⟨*proof*⟩

context *pair-qbs-s-finites*
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma
assumes [*qbs*]:$f \in X \bigotimes_Q Y \to_Q Z$
shows *qbs-bind-bind-return*:$[\![X,\alpha,\mu]\!]_{meas} \ggg= (\lambda x.\ [\![Y,\beta,\nu]\!]_{meas} \ggg= (\lambda y.$ *return-qbs* $Z\ (f\ (x,y)))) = [\![Z,\ f \circ$ (*map-prod* $\alpha\ \beta \circ$ *rr.from-real*), *distr* $(\mu \bigotimes_M \nu)$ *borel* *rr.to-real*$]\!]_{meas}$ (**is** *?lhs* = *?rhs*)
and *qbs-bind-bind-return-s-finite*: *qbs-s-finite* Z ($f \circ$ (*map-prod* $\alpha\ \beta \circ$ *rr.from-real*)) (*distr* $(\mu \bigotimes_M \nu)$ *borel rr.to-real*)
⟨*proof*⟩

end

4.1.10 The Probability Monad

definition *monadP-qbs X* ≡ *sub-qbs* (*monadM-qbs X*) {*s*. *prob-space* (*qbs-l s*)}

lemma *monadP-qbs-def2*: *monadP-qbs X* = *sub-qbs* (*all-meas-qbs X*) {*s*. *prob-space* (*qbs-l s*)}
⟨*proof*⟩

lemma
shows *qbs-space-monadPM*: $s \in$ *qbs-space* (*monadP-qbs X*) $\Longrightarrow s \in$ *qbs-space* (*monadM-qbs X*)
and *qbs-Mx-monadPM*: $f \in$ *qbs-Mx* (*monadP-qbs X*) $\Longrightarrow f \in$ *qbs-Mx* (*monadM-qbs X*)
⟨*proof*⟩

lemma *monadP-qbs-space*: *qbs-space* (*monadP-qbs X*) = {*s*. *qbs-space-of s* = X ∧ *prob-space* (*qbs-l s*)}
⟨*proof*⟩

lemma *rep-qbs-space-monadP*:
assumes $s \in$ *qbs-space* (*monadP-qbs X*)
obtains $\alpha\ \mu$ **where** $s = [\![X,\ \alpha,\ \mu]\!]_{meas}$ *qbs-prob* $X\ \alpha\ \mu$
⟨*proof*⟩

lemma *qbs-l-prob-space*:
 $s \in$ *qbs-space* (*monadP-qbs* X) $\Longrightarrow$ *prob-space* (*qbs-l* s)
 ⟨*proof*⟩

lemma *monadP-qbs-empty-iff*:
(*qbs-space* $X = \{\}$) = (*qbs-space* (*monadP-qbs* X) $= \{\}$)
⟨*proof*⟩

lemma *in-space-monadP-qbs-pred*: *qbs-pred* (*monadM-qbs* X) ($\lambda s.\ s \in$ *monadP-qbs* X)
 ⟨*proof*⟩

lemma(**in** *qbs-prob*) *in-space-monadP*[*qbs*]: $[\![X, \alpha, \mu]\!]_{meas} \in$ *qbs-space* (*monadP-qbs* X)
 ⟨*proof*⟩

lemma *qbs-morphism-monadPD*: $f \in X \to_Q$ *monadP-qbs* $Y \Longrightarrow f \in X \to_Q$ *monadM-qbs* Y
 ⟨*proof*⟩

lemma *qbs-morphism-monadPD'*: $f \in$ *monadM-qbs* $X \to_Q Y \Longrightarrow f \in$ *monadP-qbs* $X \to_Q Y$
 ⟨*proof*⟩

lemma *qbs-morphism-monadPI*:
 assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow$ *prob-space* (*qbs-l* ($f\,x$)) $f \in X \to_Q$ *monadM-qbs* Y
 shows $f \in X \to_Q$ *monadP-qbs* Y
 ⟨*proof*⟩

lemma *qbs-morphism-monadPI'*:
 assumes $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\,x \in$ *qbs-space* (*monadP-qbs* Y) $f \in X \to_Q$ *monadM-qbs* Y
 shows $f \in X \to_Q$ *monadP-qbs* Y
 ⟨*proof*⟩

lemma *qbs-morphism-monadPI''*:
 assumes $f \in$ *monadM-qbs* $X \to_Q$ *monadM-qbs* Y $\bigwedge s.\ s \in$ *qbs-space* (*monadP-qbs* X) $\Longrightarrow f\,s \in$ *qbs-space* (*monadP-qbs* Y)
 shows $f \in$ *monadP-qbs* $X \to_Q$ *monadP-qbs* Y
 ⟨*proof*⟩

lemma *monadP-qbs-Mx*: *qbs-Mx* (*monadP-qbs* X) = $\{\lambda r.\ [\![X, \alpha, k\ r]\!]_{meas}\ |\alpha\ k.\ \alpha \in$ *qbs-Mx* $X \wedge k \in$ *borel* $\to_M$ *prob-algebra borel*$\}$
⟨*proof*⟩

lemma *rep-qbs-Mx-monadP*:
 assumes $\gamma \in$ *qbs-Mx* (*monadP-qbs* X)
 obtains α k **where** $\gamma = (\lambda r.\ [\![X, \alpha, k\ r]\!]_{meas})$ $\alpha \in$ *qbs-Mx* X $k \in$ *borel* $\to_M$

prob-algebra borel $\bigwedge r$. *qbs-prob X* α (*k r*)
⟨*proof*⟩

lemma *qbs-l-monadP-le1*:*s* $\in$ *qbs-space* (*monadP-qbs X*) $\Longrightarrow$ *qbs-l s A* $\leq$ *1*
⟨*proof*⟩

lemma *qbs-l-inj-P*: *inj-on qbs-l* (*qbs-space* (*monadP-qbs X*))
⟨*proof*⟩

lemma *qbs-l-measurable-prob*[*measurable*]:*qbs-l* $\in$ *qbs-to-measure* (*monadP-qbs X*) $\rightarrow_M$ *prob-algebra* (*qbs-to-measure X*)
⟨*proof*⟩

lemma *return-qbs-morphismP*: *return-qbs X* $\in$ *X* $\rightarrow_Q$ *monadP-qbs X*
⟨*proof*⟩

lemma(**in** *qbs-prob*)
  assumes *s* = $[\![X, \alpha, \mu]\!]_{meas}$
    f $\in$ *X* $\rightarrow_Q$ *monadP-qbs Y*
    β $\in$ *qbs-Mx Y*
  and *g*[*measurable*]:*g* $\in$ *borel* $\rightarrow_M$ *prob-algebra borel*
  and ($f \circ \alpha$) = (λr. $[\![Y, \beta, g\ r]\!]_{meas}$)
  shows *bind-qbs-prob*:*qbs-prob Y* β ($\mu \gg= g$)
  and *bind-qbs'*: $s \gg= f$ = $[\![Y, \beta, \mu \gg= g]\!]_{meas}$
⟨*proof*⟩

lemma *bind-qbs-morphism'P*:
  assumes *f* $\in$ *X* $\rightarrow_Q$ *monadP-qbs Y*
  shows (λx. $x \gg= f$) $\in$ *monadP-qbs X* $\rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

lemma *distr-qbs-morphismP'*:
  assumes *f* $\in$ *X* $\rightarrow_Q$ *Y*
  shows *distr-qbs X Y f* $\in$ *monadP-qbs X* $\rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

lemma *join-qbs-morphismP*: *join-qbs* $\in$ *monadP-qbs* (*monadP-qbs X*) $\rightarrow_Q$ *monadP-qbs X*
⟨*proof*⟩

lemma
  assumes *qbs-prob* (*monadP-qbs X*) β μ
    ssx = $[\![monadP\text{-}qbs\ X, \beta, \mu]\!]_{meas}$
    α $\in$ *qbs-Mx X*
    g $\in$ *borel* $\rightarrow_M$ *prob-algebra borel*
  and β =(λr. $[\![X, \alpha, g\ r]\!]_{meas}$)
  shows *qbs-prob-join-qbs-s-finite*: *qbs-prob X* α ($\mu \gg= g$)
  and *qbs-prob-join-qbs*: *join-qbs ssx* = $[\![X, \alpha, \mu \gg= g]\!]_{meas}$
⟨*proof*⟩

context
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma *strength-qbs-ab-r-prob*:
assumes $\alpha \in$ *qbs-Mx X*
$\beta \in$ *qbs-Mx* (*monadP-qbs Y*)
$\gamma \in$ *qbs-Mx Y*
and [*measurable*]:$g \in$ *borel* $\rightarrow_M$ *prob-algebra borel*
and $\beta = (\lambda r.$ ⟦*Y*, γ, *g r*⟧$_{meas}$)
shows *qbs-prob* (*X* $\bigotimes_Q$ *Y*) (*map-prod* α γ ∘ *rr.from-real*) (*distr* (*return borel r* $\bigotimes_M$ *g r*) *borel rr.to-real*)
⟨*proof*⟩

lemma *strength-qbs-morphismP*: *strength-qbs X Y* ∈ *X* $\bigotimes_Q$ *monadP-qbs Y* $\rightarrow_Q$ *monadP-qbs* (*X* $\bigotimes_Q$ *Y*)
⟨*proof*⟩

end

lemma *bind-qbs-morphismP*: (⤜) ∈ *monadP-qbs X* $\rightarrow_Q$ (*X* $\Rightarrow_Q$ *monadP-qbs Y*) $\Rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

corollary *strength-qbs-law1P*:
assumes *x* ∈ *qbs-space* (*unit-quasi-borel* $\bigotimes_Q$ *monadP-qbs X*)
shows *snd x* = (*distr-qbs* (*unit-quasi-borel* $\bigotimes_Q$ *X*) *X snd* ∘ *strength-qbs unit-quasi-borel X*) *x*
⟨*proof*⟩

corollary *strength-qbs-law2P*:
assumes *x* ∈ *qbs-space* ((*X* $\bigotimes_Q$ *Y*) $\bigotimes_Q$ *monadP-qbs Z*)
shows (*strength-qbs X* (*Y* $\bigotimes_Q$ *Z*) ∘ (*map-prod id* (*strength-qbs Y Z*)) ∘ (λ((*x*,*y*),*z*). (*x*,(*y*,*z*)))) *x* =
(*distr-qbs* ((*X* $\bigotimes_Q$ *Y*) $\bigotimes_Q$ *Z*) (*X* $\bigotimes_Q$ (*Y* $\bigotimes_Q$ *Z*)) (λ((*x*,*y*),*z*). (*x*,(*y*,*z*))) ∘ *strength-qbs* (*X* $\bigotimes_Q$ *Y*) *Z*) *x*
⟨*proof*⟩

lemma *strength-qbs-law4P*:
assumes *x* ∈ *qbs-space* (*X* $\bigotimes_Q$ *monadP-qbs* (*monadP-qbs Y*))
shows (*strength-qbs X Y* ∘ *map-prod id join-qbs*) *x* = (*join-qbs* ∘ *distr-qbs* (*X* $\bigotimes_Q$ *monadP-qbs Y*) (*monadP-qbs* (*X* $\bigotimes_Q$ *Y*)) (*strength-qbs X Y*) ∘ *strength-qbs X* (*monadP-qbs Y*)) *x*
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *distr-qbs-morphismP*: *distr-qbs X Y* $\in X \Rightarrow_Q Y \rightarrow_Q$ *monadP-qbs X* $\Rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

lemma *bind-qbs-return-rotateP*:
assumes $p \in$ *qbs-space* (*monadP-qbs X*)
and $q \in$ *qbs-space* (*monadP-qbs Y*)
shows $q \gg= (\lambda y.\ p \gg= (\lambda x.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y))) = p \gg= (\lambda x.\ q \gg= (\lambda y.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y)))$
⟨*proof*⟩

lemma *qbs-bind-bind-return1P*:
assumes $f \in X \bigotimes_Q Y \rightarrow_Q$ *monadP-qbs Z*
$p \in$ *qbs-space* (*monadP-qbs X*)
$q \in$ *qbs-space* (*monadP-qbs Y*)
shows $q \gg= (\lambda y.\ p \gg= (\lambda x.\ f\ (x,y))) = (q \gg= (\lambda y.\ p \gg= (\lambda x.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y)))) \gg= f$
⟨*proof*⟩

corollary *qbs-bind-bind-return1P'*:
assumes [*qbs*]:$f \in$ *qbs-space* $(X \Rightarrow_Q Y \Rightarrow_Q$ *monadP-qbs Z*)
$p \in$ *qbs-space* (*monadP-qbs X*)
$q \in$ *qbs-space* (*monadP-qbs Y*)
shows $q \gg= (\lambda y.\ p \gg= (\lambda x.\ f\ x\ y)) = (q \gg= (\lambda y.\ p \gg= (\lambda x.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y)))) \gg=$ (*case-prod f*)
⟨*proof*⟩

lemma *qbs-bind-bind-return2P*:
assumes $f \in X \bigotimes_Q Y \rightarrow_Q$ *monadP-qbs Z*
$p \in$ *qbs-space* (*monadP-qbs X*) $q \in$ *qbs-space* (*monadP-qbs Y*)
shows $p \gg= (\lambda x.\ q \gg= (\lambda y.\ f\ (x,y))) = (p \gg= (\lambda x.\ q \gg= (\lambda y.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y)))) \gg= f$
⟨*proof*⟩

corollary *qbs-bind-bind-return2P'*:
assumes [*qbs*]:$f \in$ *qbs-space* $(X \Rightarrow_Q Y \Rightarrow_Q$ *monadP-qbs Z*)
$p \in$ *qbs-space* (*monadP-qbs X*)
$q \in$ *qbs-space* (*monadP-qbs Y*)
shows $p \gg= (\lambda x.\ q \gg= (\lambda y.\ f\ x\ y)) = (p \gg= (\lambda x.\ q \gg= (\lambda y.$ *return-qbs* $(X \bigotimes_Q Y)\ (x,y)))) \gg=$ (*case-prod f*)
⟨*proof*⟩

corollary *bind-qbs-rotateP*:
assumes $f \in X \bigotimes_Q Y \rightarrow_Q$ *monadP-qbs Z*
$p \in$ *qbs-space* (*monadP-qbs X*)
and $q \in$ *qbs-space* (*monadP-qbs Y*)
shows $q \gg= (\lambda y.\ p \gg= (\lambda x.\ f\ (x,y))) = p \gg= (\lambda x.\ q \gg= (\lambda y.\ f\ (x,y)))$
⟨*proof*⟩

context *pair-qbs-probs*
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

sublocale *qbs-prob X* $\bigotimes_Q$ *Y map-prod* α β ∘ *rr.from-real distr* (μ $\bigotimes_M$ ν) *borel rr.to-real*
⟨*proof*⟩

lemma *qbs-bind-bind-return-prob*:
assumes [*qbs*]:*f* ∈ *X* $\bigotimes_Q$ *Y* $\rightarrow_Q$ *Z*
shows *qbs-prob Z* (*f* ∘ (*map-prod* α β ∘ *rr.from-real*)) (*distr* (μ $\bigotimes_M$ ν) *borel rr.to-real*)
⟨*proof*⟩

end

4.1.11 Almost Everywhere

lift-definition *qbs-almost-everywhere* :: [*'a qbs-measure*, *'a* ⇒ *bool*] ⇒ *bool*
is λ(*X*,α,μ). *almost-everywhere* (*distr* μ (*qbs-to-measure X*) α)
⟨*proof*⟩

syntax
-qbs-almost-everywhere :: *pttrn* ⇒ *'a* ⇒ *bool* ⇒ *bool* (AE_Q *- in -. -* [*0*,*0*,*10*] *10*)

syntax-consts
-qbs-almost-everywhere ⇌ *qbs-almost-everywhere*

translations
AE_Q *x in p. P* ⇌ *CONST qbs-almost-everywhere p* (λ*x. P*)

lemma *AEq-qbs-l*: (AE_Q *x in p. P x*) = (*AE x in qbs-l p. P x*)
⟨*proof*⟩

lemma(**in** *qbs-meas*) *AEq-def*:
(AE_Q *x in* $[\![X, \alpha, \mu]\!]_{meas}$. *P x*) = (*AE x in* (*distr* μ (*qbs-to-measure X*) α). *P x*)
⟨*proof*⟩

lemma(**in** *qbs-meas*) *AEq-AE*: (AE_Q *x in* $[\![X, \alpha, \mu]\!]_{meas}$. *P x*) ⟹ (*AE x in* μ. *P* (α *x*))
⟨*proof*⟩

lemma(**in** *qbs-meas*) *AEq-AE-iff*:
assumes [*qbs*]:*qbs-pred X P*
shows (AE_Q *x in* $[\![X, \alpha, \mu]\!]_{meas}$. *P x*) ⟷ (*AE x in* μ. *P* (α *x*))
⟨*proof*⟩

lemma *AEq-qbs-pred*[*qbs*]: *qbs-almost-everywhere* $\in$ *monadM-qbs X* $\to_Q$ (*X* $\Rightarrow_Q$ *qbs-count-space UNIV*) $\Rightarrow_Q$ *qbs-count-space UNIV*
⟨*proof*⟩

lemma *AEq-I2'*[*simp*]:
assumes *p* $\in$ *qbs-space* (*all-meas-qbs X*) $\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow P\ x$
shows AE_Q *x in p. P x*
⟨*proof*⟩

lemma *AEq-I2*[*simp*]:
assumes *p* $\in$ *qbs-space* (*monadM-qbs X*) $\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow P\ x$
shows AE_Q *x in p. P x*
⟨*proof*⟩

lemma *AEq-mp*[*elim*!]:
assumes AE_Q *x in s. P x* AE_Q *x in s. P x* $\longrightarrow Q\ x$
shows AE_Q *x in s. Q x*
⟨*proof*⟩

lemma
shows *AEq-iffI*: AE_Q *x in s. P x* $\Longrightarrow AE_Q$ *x in s. P x* $\longleftrightarrow Q\ x \Longrightarrow AE_Q$ *x in s. Q x*
and *AEq-disjI1*: AE_Q *x in s. P x* $\Longrightarrow AE_Q$ *x in s. P x* $\vee$ *Q x*
and *AEq-disjI2*: AE_Q *x in s. Q x* $\Longrightarrow AE_Q$ *x in s. P x* $\vee$ *Q x*
and *AEq-conjI*: AE_Q *x in s. P x* $\Longrightarrow AE_Q$ *x in s. Q x* $\Longrightarrow AE_Q$ *x in s. P x* $\wedge$ *Q x*
and *AEq-conj-iff*[*simp*]: (AE_Q *x in s. P x* $\wedge$ *Q x*) $\longleftrightarrow$ (AE_Q *x in s. P x*) $\wedge$ (AE_Q *x in s. Q x*)
⟨*proof*⟩

lemma *AEq-symmetric*:
assumes AE_Q *x in s. P x = Q x*
shows AE_Q *x in s. Q x = P x*
⟨*proof*⟩

lemma *AEq-impI*: (*P* $\Longrightarrow AE_Q$ *x in M. Q x*) $\Longrightarrow AE_Q$ *x in M. P* $\longrightarrow Q\ x$
⟨*proof*⟩

lemma
shows *AEq-Ball-mp-all-meas*:
s $\in$ *qbs-space* (*all-meas-qbs X*) $\Longrightarrow$ ($\bigwedge x.\ x\in$*qbs-space X* $\Longrightarrow P\ x$) $\Longrightarrow AE_Q$ *x in s. P x* $\longrightarrow Q\ x \Longrightarrow AE_Q$ *x in s. Q x*
and *AEq-Ball-mp*:
s $\in$ *qbs-space* (*monadM-qbs X*) $\Longrightarrow$ ($\bigwedge x.\ x\in$*qbs-space X* $\Longrightarrow P\ x$) $\Longrightarrow AE_Q$ *x in s. P x* $\longrightarrow Q\ x \Longrightarrow AE_Q$ *x in s. Q x*
and *AEq-cong-all-meas*:
s $\in$ *qbs-space* (*all-meas-qbs X*) $\Longrightarrow$ ($\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow P\ x \longleftrightarrow Q\ x$) $\Longrightarrow$ (AE_Q *x in s. P x*) $\longleftrightarrow$ (AE_Q *x in s. Q x*)

and *AEq-cong*:
$s \in$ *qbs-space* (*monadM-qbs* X) $\Longrightarrow$ ($\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow P\ x \longleftrightarrow Q\ x$) $\Longrightarrow$ ($AE_Q\ x$ *in* $s.\ P\ x$) $\longleftrightarrow$ ($AE_Q\ x$ *in* $s.\ Q\ x$)
⟨*proof*⟩

lemma
shows *AEq-cong-simp-all-meas*: $s \in$ *qbs-space* (*all-meas-qbs* X) $\Longrightarrow$ ($\bigwedge x.\ x \in$ *qbs-space* X =*simp*=> $P\ x = Q\ x$) $\Longrightarrow$ ($AE_Q\ x$ *in* $s.\ P\ x$) $\longleftrightarrow$ ($AE_Q\ x$ *in* $s.\ Q\ x$)
and *AEq-cong-simp*: $s \in$ *qbs-space* (*monadM-qbs* X) $\Longrightarrow$ ($\bigwedge x.\ x \in$ *qbs-space* X =*simp*=> $P\ x = Q\ x$) $\Longrightarrow$ ($AE_Q\ x$ *in* $s.\ P\ x$) $\longleftrightarrow$ ($AE_Q\ x$ *in* $s.\ Q\ x$)
⟨*proof*⟩

lemma *AEq-all-countable*: ($AE_Q\ x$ *in* $s.\ \forall i.\ P\ i\ x$) $\longleftrightarrow$ ($\forall i$::'*i*::*countable*. $AE_Q\ x$ *in* $s.\ P\ i\ x$)
⟨*proof*⟩

lemma *AEq-ball-countable*: *countable* $X \Longrightarrow$ ($AE_Q\ x$ *in* $s.\ \forall y{\in}X.\ P\ x\ y$) $\longleftrightarrow$ ($\forall y{\in}X.\ AE_Q\ x$ *in* $s.\ P\ x\ y$)
⟨*proof*⟩

lemma *AEq-ball-countable'*: ($\bigwedge N.\ N \in I \Longrightarrow AE_Q\ x$ *in* $s.\ P\ N\ x$) $\Longrightarrow$ *countable* $I \Longrightarrow AE_Q\ x$ *in* $s.\ \forall N \in I.\ P\ N\ x$
⟨*proof*⟩

lemma *AEq-pairwise*: *countable* $F \Longrightarrow$ *pairwise* ($\lambda A\ B.\ AE_Q\ x$ *in* $s.\ R\ x\ A\ B$) F $\longleftrightarrow$ ($AE_Q\ x$ *in* $s.$ *pairwise* ($R\ x$) F)
⟨*proof*⟩

lemma *AEq-finite-all*: *finite* $S \Longrightarrow$ ($AE_Q\ x$ *in* $s.\ \forall i{\in}S.\ P\ i\ x$) $\longleftrightarrow$ ($\forall i{\in}S.\ AE_Q\ x$ *in* $s.\ P\ i\ x$)
⟨*proof*⟩

lemma *AEq-finite-allI*:*finite* $S \Longrightarrow$ ($\bigwedge s.\ s \in S \Longrightarrow AE_Q\ x$ *in* $M.\ Q\ s\ x$) $\Longrightarrow AE_Q\ x$ *in* $M.\ \forall s{\in}S.\ Q\ s\ x$
⟨*proof*⟩

4.1.12 Integral

lift-definition *qbs-nn-integral* :: ['*a qbs-measure*, '*a* ⇒ *ennreal*] ⇒ *ennreal*
is $\lambda(X,\alpha,\mu)\ f.(\int^+ x.\ f\ x\ \partial$*distr* μ (*qbs-to-measure* X) α)
⟨*proof*⟩

lift-definition *qbs-integral* :: ['*a qbs-measure*, '*a* ⇒ ('*b* :: {*banach*,*second-countable-topology*})] ⇒ '*b*
is $\lambda(X,\alpha,\mu)\ f.$ *if* $f \in X \rightarrow_Q$ *qbs-borel then* ($\int x.\ f\ (\alpha\ x)\ \partial\mu$) *else 0*
⟨*proof*⟩

syntax
-*qbs-nn-integral* :: *pttrn* ⇒ *ennreal* ⇒ '*a qbs-measure* ⇒ *ennreal* ($\int^+{}_Q$((*2* -./ -)/

∂-) [60,61] 110)

syntax-consts
-qbs-nn-integral $\rightleftharpoons$ *qbs-nn-integral*

translations
$\int^{+}{}_{Q}$ *x. f* ∂p $\rightleftharpoons$ *CONST qbs-nn-integral p* ($\lambda x.$ *f*)

syntax
-qbs-integral :: *pttrn* $\Rightarrow$ - $\Rightarrow$ *'a qbs-measure* $\Rightarrow$ - ($\int_{Q}$((2 -./ -)/ ∂-) [60,61] 110)

syntax-consts
-qbs-integral $\rightleftharpoons$ *qbs-integral*

translations
$\int_{Q}$ *x. f* ∂p $\rightleftharpoons$ *CONST qbs-integral p* ($\lambda x.$ *f*)

lemma(**in** *qbs-meas*)
shows *qbs-nn-integral-def*: $f \in X \rightarrow_Q$ *qbs-borel* $\Longrightarrow$ ($\int^{+}{}_{Q}$ *x. f x* $\partial[\![X, \alpha, \mu]\!]_{meas}$) = ($\int^{+}x.$ *f* (α *x*) ∂ μ)
and *qbs-nn-integral-def2*:($\int^{+}{}_{Q}$ *x. f x* $\partial[\![X, \alpha, \mu]\!]_{meas}$) = ($\int^{+}x.$ *f x* ∂(*distr* μ (*qbs-to-measure X*) α))
⟨*proof*⟩

lemma(**in** *qbs-meas*) *qbs-integral-def*:
$f \in X \rightarrow_Q$ *qbs-borel* $\Longrightarrow$ ($\int_{Q}$ *x. f x* $\partial[\![X, \alpha, \mu]\!]_{meas}$) = ($\int x.$ *f* (α *x*) ∂ μ)
⟨*proof*⟩

lemma(**in** *qbs-meas*) *qbs-integral-def2*: ($\int_{Q}$ *x. f x* $\partial[\![X, \alpha, \mu]\!]_{meas}$) = ($\int x.$ *f x* ∂(*distr* μ (*qbs-to-measure X*) α))
⟨*proof*⟩

lemma *qbs-measure-eqI-all-meas*:
assumes [*qbs*]:*p* $\in$ *qbs-space* (*all-meas-qbs X*) *q* $\in$ *qbs-space* (*all-meas-qbs X*)
and $\bigwedge f.$ $f \in X \rightarrow_Q$ *qbs-borel* $\Longrightarrow$ ($\int^{+}{}_{Q}$ *x. f x* ∂p) = ($\int^{+}{}_{Q}$ *x. f x* ∂q)
shows *p* = *q*
⟨*proof*⟩

lemma *qbs-measure-eqI*:
assumes [*qbs*]:*p* $\in$ *qbs-space* (*monadM-qbs X*) *q* $\in$ *qbs-space* (*monadM-qbs X*)
and $\bigwedge f.$ $f \in X \rightarrow_Q$ *qbs-borel* $\Longrightarrow$ ($\int^{+}{}_{Q}$ *x. f x* ∂p) = ($\int^{+}{}_{Q}$ *x. f x* ∂q)
shows *p* = *q*
⟨*proof*⟩

lemma *qbs-nn-integral-def2-l*: *qbs-nn-integral s f* = $integral^{N}$ (*qbs-l s*) *f*
⟨*proof*⟩

lemma *qbs-integral-def2-l*: *qbs-integral s f* = $integral^{L}$ (*qbs-l s*) *f*
⟨*proof*⟩

definition *qbs-integrable* :: *'a qbs-measure* ⇒ (*'a* ⇒ *'b*::{*second-countable-topology*,*real-normed-vector*}) ⇒ *bool*
where *qbs-integrable-iff-integrable*: *qbs-integrable p f* ⟷ *integrable* (*qbs-l p*) *f*

lemma(**in** *qbs-meas*) *qbs-integrable-def*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
shows *qbs-integrable* $[\![X,\ \alpha,\ \mu]\!]_{meas}$ *f* ⟷ *f* ∈ *X* $\to_Q$ *qbs-borel* ∧ *integrable* *μ* (*λx*. *f* (*α x*))
⟨*proof*⟩

lemma *qbs-integrable-morphism-dest-all-meas*:
assumes *s* ∈ *qbs-space* (*all-meas-qbs X*)
and *qbs-integrable s f*
shows *f* ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-integrable-morphism-dest*:
assumes *s* ∈ *qbs-space* (*monadM-qbs X*)
and *qbs-integrable s f*
shows *f* ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma *qbs-integrable-morphismP*:
assumes *s* ∈ *qbs-space* (*monadP-qbs X*)
and *qbs-integrable s f*
shows *f* ∈ *X* $\to_Q$ *qbs-borel*
⟨*proof*⟩

lemma(**in** *qbs-s-finite*) *qbs-integrable-measurable*[*simp*]:
assumes *qbs-integrable* $[\![X,\alpha,\mu]\!]_{meas}$ *f*
shows *f* ∈ *qbs-to-measure X* $\to_M$ *borel*
⟨*proof*⟩

corollary(**in** *qbs-meas*) *qbs-integrable-distr*: *qbs-integrable* $[\![X,\ \alpha,\ \mu]\!]_{meas}$ *f* = *integrable* (*distr μ* (*qbs-to-measure X*) *α*) *f*
⟨*proof*⟩

lemma *qbs-integrable-morphism*[*qbs*]: *qbs-integrable* ∈ *monadM-qbs X* $\to_Q$ (*X* $\Rightarrow_Q$ (*qbs-borel* :: (*'a* :: {*banach*, *second-countable-topology*}) *quasi-borel*)) $\Rightarrow_Q$ *qbs-count-space UNIV*
⟨*proof*⟩

lemma(**in** *qbs-meas*) *qbs-integrable-iff-integrable*:
assumes *f* ∈ *qbs-to-measure X* $\to_M$ (*borel* :: - ::{*second-countable-topology*,*banach*} *measure*)
shows *qbs-integrable* $[\![X,\ \alpha,\ \mu]\!]_{meas}$ *f* = *integrable* *μ* (*λx*. *f* (*α x*))
⟨*proof*⟩

lemma *qbs-integrable-iff-bounded-all-meas*:
fixes f :: $'a \Rightarrow 'b$::{*second-countable-topology*,*banach*}
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
shows *qbs-integrable s f* $\longleftrightarrow f \in X \rightarrow_Q$ *qbs-borel* $\wedge$ ($\int^{+}{}_Q$ *x. ennreal* (*norm* (*f x*)) ∂s) $< \infty$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemmas *qbs-integrable-iff-bounded* = *qbs-integrable-iff-bounded-all-meas*[*OF monadM-all-meas-space*]

lemma *not-qbs-integrable-qbs-integral*: ¬ *qbs-integrable s f* $\Longrightarrow$ *qbs-integral s f* = *0*
⟨*proof*⟩

lemma *qbs-integrable-cong-AE-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
AE_Q *x in s. f x* = *g x*
and *qbs-integrable s f* $g \in X \rightarrow_Q$ *qbs-borel*
shows *qbs-integrable s g*
⟨*proof*⟩

lemmas *qbs-integrable-cong-AE* = *qbs-integrable-cong-AE-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integrable-cong-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
$\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow$ *f x* = *g x*
and *qbs-integrable s f*
shows *qbs-integrable s g*
⟨*proof*⟩

lemmas *qbs-integrable-cong* = *qbs-integrable-cong-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integrable-zero*[*simp*, *intro*]: *qbs-integrable s* ($\lambda x.$ *0*)
⟨*proof*⟩

lemma *qbs-integrable-const*:
assumes $s \in$ *qbs-space* (*monadP-qbs X*)
shows *qbs-integrable s* ($\lambda x.$ *c*)
⟨*proof*⟩

lemma *qbs-integrable-add*[*simp*,*intro*!]:
assumes *qbs-integrable s f*
and *qbs-integrable s g*
shows *qbs-integrable s* ($\lambda x.$ *f x* + *g x*)
⟨*proof*⟩

lemma *qbs-integrable-diff*[*simp*,*intro*!]:
assumes *qbs-integrable s f*

and *qbs-integrable s g*
shows *qbs-integrable s* ($\lambda x.$ *f x* $-$ *g x*)
⟨*proof*⟩

lemma *qbs-integrable-sum*[*simp*, *intro*!]: ($\bigwedge i.$ $i \in I \Longrightarrow$ *qbs-integrable s* (*f i*)) $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ $\sum i{\in}I.$ *f i x*)
⟨*proof*⟩

lemma *qbs-integrable-scaleR-left*[*simp*, *intro*!]: *qbs-integrable s f* $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *f x* $*_R$ (*c* :: *'a* :: {*second-countable-topology*,*banach*}))
⟨*proof*⟩

lemma *qbs-integrable-scaleR-right*[*simp*, *intro*!]: *qbs-integrable s f* $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *c* $*_R$ (*f x* :: *'a* :: {*second-countable-topology*,*banach*}))
⟨*proof*⟩

lemma *qbs-integrable-mult-iff*:
fixes *f* :: *'a* ⇒ *real*
shows (*qbs-integrable s* ($\lambda x.$ *c* $*$ *f x*)) = (*c* = *0* ∨ *qbs-integrable s f*)
⟨*proof*⟩

lemma
fixes *c* :: -::{*real-normed-algebra*,*second-countable-topology*}
assumes *qbs-integrable s f*
shows *qbs-integrable-mult-right*:*qbs-integrable s* ($\lambda x.$ *c* $*$ *f x*)
and *qbs-integrable-mult-left*: *qbs-integrable s* ($\lambda x.$ *f x* $*$ *c*)
⟨*proof*⟩

lemma *qbs-integrable-divide-zero*[*simp*, *intro*!]:
fixes *c* :: -::{*real-normed-field*, *field*, *second-countable-topology*}
shows *qbs-integrable s f* $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *f x* / *c*)
⟨*proof*⟩

lemma *qbs-integrable-inner-left*[*simp*, *intro*!]:
qbs-integrable s f $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *f x* · *c*)
⟨*proof*⟩

lemma *qbs-integrable-inner-right*[*simp*, *intro*!]:
qbs-integrable s f $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *c* · *f x*)
⟨*proof*⟩

lemma *qbs-integrable-minus*[*simp*, *intro*!]:
qbs-integrable s f $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ $-$ *f x*)
⟨*proof*⟩

lemma [*simp*, *intro*]:
assumes *qbs-integrable s f*
shows *qbs-integrable-Re*: *qbs-integrable s* ($\lambda x.$ *Re* (*f x*))
and *qbs-integrable-Im*: *qbs-integrable s* ($\lambda x.$ *Im* (*f x*))

and *qbs-integrable-cnj*: *qbs-integrable s* ($\lambda x.$ *cnj* (*f x*))
⟨*proof*⟩

lemma *qbs-integrable-of-real*[*simp*, *intro*!]: *qbs-integrable s f* $\Longrightarrow$ *qbs-integrable s* ($\lambda x.$ *of-real* (*f x*))
⟨*proof*⟩

lemma [*simp*, *intro*]:
assumes *qbs-integrable s f*
shows *qbs-integrable-fst*: *qbs-integrable s* ($\lambda x.$ *fst* (*f x*))
and *qbs-integrable-snd*: *qbs-integrable s* ($\lambda x.$ *snd* (*f x*))
⟨*proof*⟩

lemma *qbs-integrable-norm*:
fixes *f* :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes *qbs-integrable s f*
shows *qbs-integrable s* ($\lambda x.$ *norm* (*f x*))
⟨*proof*⟩

lemma *qbs-integrable-abs*:
fixes *f* :: - ⇒ *real*
assumes *qbs-integrable s f*
shows *qbs-integrable s* ($\lambda x.$ |*f x*|)
⟨*proof*⟩

lemma *qbs-integrable-sq*:
fixes *c* :: -::{*real-normed-field*,*second-countable-topology*}
assumes *qbs-integrable s* ($\lambda x.$ *c*) *qbs-integrable s f*
and *qbs-integrable s* ($\lambda x.$ $(f\ x)^2$)
shows *qbs-integrable s* ($\lambda x.$ $(f\ x - c)^2$)
⟨*proof*⟩

lemma *qbs-nn-integral-eq-integral-AEq*:
assumes *qbs-integrable s f* AE_Q *x in s.* $0 \le f\ x$
shows ($\int^{+}{}_Q$ *x. ennreal* (*f x*) ∂s) = *ennreal* ($\int_Q$ *x. f x* ∂s)
⟨*proof*⟩

lemma *qbs-nn-integral-eq-integral-all-meas*:
assumes *s* ∈ *qbs-space* (*all-meas-qbs X*) *qbs-integrable s f*
and ⋀*x. x* ∈ *qbs-space X* $\Longrightarrow$ $0 \le f\ x$
shows ($\int^{+}{}_Q$ *x. ennreal* (*f x*) ∂s) = *ennreal* ($\int_Q$ *x. f x* ∂s)
⟨*proof*⟩

lemmas *qbs-nn-integral-eq-integral* = *qbs-nn-integral-eq-integral-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-nn-integral-cong-AEq-all-meas*:
assumes *s* ∈ *qbs-space* (*all-meas-qbs X*) AE_Q *x in s. f x* = *g x*
shows *qbs-nn-integral s f* = *qbs-nn-integral s g*

⟨*proof*⟩

lemmas *qbs-nn-integral-cong-AEq* = *qbs-nn-integral-cong-AEq-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-nn-integral-cong-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*) $\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow f\ x = g\ x$
shows *qbs-nn-integral s f* = *qbs-nn-integral s g*
⟨*proof*⟩

lemmas *qbs-nn-integral-cong* = *qbs-nn-integral-cong-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-nn-integral-const*:
$(\int^{+}_{Q} x.\ c\ \partial s) = c * $ *qbs-l s* (*qbs-space* (*qbs-space-of s*))
⟨*proof*⟩

lemma *qbs-nn-integral-const-prob*:
assumes $s \in$ *qbs-space* (*monadP-qbs X*)
shows $(\int^{+}_{Q} x.\ c\ \partial s) = c$
⟨*proof*⟩

lemma *qbs-nn-integral-add-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*)
and [*qbs*]:$f \in X \to_Q$ *qbs-borel* $g \in X \to_Q$ *qbs-borel*
shows $(\int^{+}_{Q} x.\ f\ x + g\ x\ \partial s) = (\int^{+}_{Q} x.\ f\ x\ \partial s) + (\int^{+}_{Q} x.\ g\ x\ \partial s)$
⟨*proof*⟩

lemmas *qbs-nn-integral-add* = *qbs-nn-integral-add-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-nn-integral-cmult-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*) **and** [*qbs*]:$f \in X \to_Q$ *qbs-borel*
shows $(\int^{+}_{Q} x.\ c * f\ x\ \partial s) = c * (\int^{+}_{Q} x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemmas *qbs-nn-integral-cmult* = *qbs-nn-integral-cmult-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-cong-AEq-all-meas*:
assumes [*qbs*]:$s \in$ *qbs-space* (*all-meas-qbs X*) $f \in X \to_Q$ *qbs-borel* $g \in X \to_Q$ *qbs-borel*
and $AE_Q\ x$ *in s*. $f\ x = g\ x$
shows *qbs-integral s f* = *qbs-integral s g*
⟨*proof*⟩

lemmas *qbs-integral-cong-AEq* = *qbs-integral-cong-AEq-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-cong-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs X*) $\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow f\ x = g\ x$
shows *qbs-integral s f* = *qbs-integral s g*
⟨*proof*⟩

lemmas *qbs-integral-cong* = *qbs-integral-cong-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-nonneg-AEq*:
 fixes *f* :: - ⇒ *real*
 shows AE_Q *x in s. 0* ≤ *f x* ⟹ *0* ≤ *qbs-integral s f*
 ⟨*proof*⟩

lemma *qbs-integral-nonneg-all-meas*:
 fixes *f* :: - ⇒ *real*
 assumes *s* ∈ *qbs-space* (*all-meas-qbs X*) ⋀*x. x* ∈ *qbs-space X* ⟹ *0* ≤ *f x*
 shows *0* ≤ *qbs-integral s f*
 ⟨*proof*⟩

lemmas *qbs-integral-nonneg* = *qbs-integral-nonneg-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-mono-AEq*:
 fixes *f* :: - ⇒ *real*
 assumes *qbs-integrable s f qbs-integrable s g* AE_Q *x in s. f x* ≤ *g x*
  shows *qbs-integral s f* ≤ *qbs-integral s g*
 ⟨*proof*⟩

lemma *qbs-integral-mono-all-meas*:
 fixes *f* :: - ⇒ *real*
 assumes *s* ∈ *qbs-space* (*all-meas-qbs X*)
   and *qbs-integrable s f qbs-integrable s g* ⋀*x. x* ∈ *qbs-space X* ⟹ *f x* ≤ *g x*
  shows *qbs-integral s f* ≤ *qbs-integral s g*
 ⟨*proof*⟩

lemmas *qbs-integral-mono* = *qbs-integral-mono-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-const-prob*:
 assumes *s* ∈ *qbs-space* (*monadP-qbs X*)
 shows ($\int_Q$ *x. c* ∂*s*) = *c*
 ⟨*proof*⟩

lemma
 assumes *qbs-integrable s f qbs-integrable s g*
 shows *qbs-integral-add*:($\int_Q$ *x. f x* + *g x* ∂*s*) = ($\int_Q$ *x. f x* ∂*s*) + ($\int_Q$ *x. g x* ∂*s*)
  and *qbs-integral-diff*: ($\int_Q$ *x. f x* − *g x* ∂*s*) = ($\int_Q$ *x. f x* ∂*s*) − ($\int_Q$ *x. g x* ∂*s*)
 ⟨*proof*⟩

lemma [*simp*]:
 fixes *c* :: -::{*real-normed-field*,*second-countable-topology*}
 shows *qbs-integral-mult-right-zero*:($\int_Q$ *x. c* ∗ *f x* ∂*s*) = *c* ∗ ($\int_Q$ *x. f x* ∂*s*)
  and *qbs-integral-mult-left-zero*:($\int_Q$ *x. f x* ∗ *c* ∂*s*) = ($\int_Q$ *x. f x* ∂*s*) ∗ *c*
  and *qbs-integral-divide-zero*: ($\int_Q$ *x. f x* / *c* ∂*s*) = ($\int_Q$ *x. f x* ∂*s*) / *c*
  ⟨*proof*⟩

lemma *qbs-integral-minus*[*simp*]: $(\int_Q x.\ -\ f\ x\ \partial s) = -\ (\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma [*simp*]:
shows *qbs-integral-scaleR-right*:$(\int_Q x.\ c *_R f\ x\ \partial s) = c *_R (\int_Q x.\ f\ x\ \partial s)$
and *qbs-integral-scaleR-left*: $(\int_Q x.\ f\ x *_R c\ \partial s) = (\int_Q x.\ f\ x\ \partial s) *_R c$
⟨*proof*⟩

lemma [*simp*]:
shows *qbs-integral-inner-left*: *qbs-integrable s f* $\Longrightarrow (\int_Q x.\ f\ x \cdot c\ \partial s) = (\int_Q x.\ f\ x\ \partial s) \cdot c$
and *qbs-integral-inner-right*: *qbs-integrable s f* $\Longrightarrow (\int_Q x.\ c \cdot f\ x\ \partial s) = c \cdot (\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma *integral-complex-of-real*[*simp*]: $(\int_Q x.$ *complex-of-real* $(f\ x)\ \partial s)=$ *of-real* $(\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma *integral-cnj*[*simp*]: $(\int_Q x.$ *cnj* $(f\ x)\ \partial s) =$ *cnj* $(\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma [*simp*]:
assumes *qbs-integrable s f*
shows *qbs-integral-Im*: $(\int_Q x.$ *Im* $(f\ x)\ \partial s) =$ *Im* $(\int_Q x.\ f\ x\ \partial s)$
and *qbs-integral-Re*: $(\int_Q x.$ *Re* $(f\ x)\ \partial s) =$ *Re* $(\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma *qbs-integral-of-real*[*simp*]:*qbs-integrable s f* $\Longrightarrow (\int_Q x.$ *of-real* $(f\ x)\ \partial s) =$ *of-real* $(\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma [*simp*]:
assumes *qbs-integrable s f*
shows *qbs-integral-fst*: $(\int_Q x.$ *fst* $(f\ x)\ \partial s) =$ *fst* $(\int_Q x.\ f\ x\ \partial s)$
and *qbs-integral-snd*: $(\int_Q x.$ *snd* $(f\ x)\ \partial s) =$ *snd* $(\int_Q x.\ f\ x\ \partial s)$
⟨*proof*⟩

lemma *real-qbs-integral-def*:
assumes *qbs-integrable s f*
shows *qbs-integral s f* = *enn2real* $(\int^+_Q x.$ *ennreal* $(f\ x)\ \partial s) -$ *enn2real* $(\int^+_Q x.$ *ennreal* $(-\ f\ x)\ \partial s)$
⟨*proof*⟩

lemma *Markov-inequality-qbs-prob*:
qbs-integrable s f $\Longrightarrow AE_Q$ *x in s*. $0 \le f\ x \Longrightarrow 0 < c \Longrightarrow \mathcal{P}(x$ *in qbs-l s*. $c \le f\ x) \le (\int_Q x.\ f\ x\ \partial s)\ /\ c$
⟨*proof*⟩

lemma *Chebyshev-inequality-qbs-prob*:
assumes $s \in$ *qbs-space* (*monadP-qbs X*)
and $f \in X \rightarrow_Q$ *qbs-borel qbs-integrable s* $(\lambda x.\ (f\ x)^2)$
and $0 < e$
shows $\mathcal{P}(x\ in\ qbs\text{-}l\ s.\ e \leq |f\ x - (\int_Q x.\ f\ x\ \partial s)|) \leq (\int_Q x.\ (f\ x - (\int_Q x.\ f\ x\ \partial s))^2\ \partial s)\ /\ e^2$
⟨*proof*⟩

lemma *qbs-l-return-qbs*:
assumes $x \in$ *qbs-space X*
shows *qbs-l* (*return-qbs X x*) = *return* (*qbs-to-measure X*) *x*
⟨*proof*⟩

lemma *qbs-l-bind-qbs-all-meas*:
assumes [*qbs*]: $s \in$ *qbs-space* (*all-meas-qbs X*) $f \in X \rightarrow_Q$ *all-meas-qbs Y*
shows *qbs-l* ($s \gg= f$) = *qbs-l s* $\gg=_k$ *qbs-l* $\circ f$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-l-bind-qbs*:
assumes $s \in$ *qbs-space* (*monadM-qbs X*) $f \in X \rightarrow_Q$ *monadM-qbs Y*
shows *qbs-l* ($s \gg= f$) = *qbs-l s* $\gg=_k$ *qbs-l* $\circ f$
⟨*proof*⟩

lemma *qbs-l-bind-qbsP*:
assumes [*qbs*]: $s \in$ *qbs-space* (*monadP-qbs X*) $f \in X \rightarrow_Q$ *monadP-qbs Y*
shows *qbs-l* ($s \gg= f$) = *qbs-l s* $\gg=$ *qbs-l* $\circ f$
⟨*proof*⟩

lemma *qbs-integrable-return*[*simp*, *intro*]:
fixes f :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes $x \in$ *qbs-space X* $f \in X \rightarrow_Q$ *qbs-borel*
shows *qbs-integrable* (*return-qbs X x*) *f*
⟨*proof*⟩

lemma *qbs-integrable-bind-return-all-meas*:
fixes f :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes [*qbs*]:$s \in$ *qbs-space* (*all-meas-qbs X*) $f \in Y \rightarrow_Q$ *qbs-borel* $g \in X \rightarrow_Q Y$
shows *qbs-integrable* ($s \gg=$ ($\lambda x.$ *return-qbs Y* ($g\ x$))) *f* = *qbs-integrable s* ($f \circ g$) (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-integrable-bind-return*:
fixes f :: *'a* ⇒ *'b*::{*second-countable-topology*,*banach*}
assumes $s \in$ *qbs-space* (*monadM-qbs X*) $f \in Y \rightarrow_Q$ *qbs-borel* $g \in X \rightarrow_Q Y$
shows *qbs-integrable* ($s \gg=$ ($\lambda x.$ *return-qbs Y* ($g\ x$))) *f* = *qbs-integrable s* ($f \circ g$)
⟨*proof*⟩

lemma *qbs-nn-integral-morphism*[*qbs*]: *qbs-nn-integral* $\in$ *monadM-qbs X* $\rightarrow_Q$ ($X \Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ *qbs-borel*

⟨proof⟩

lemma *qbs-nn-integral-morphism′*:
assumes [*qbs*,*measurable*]:*f* ∈ *X* $\to_Q$ *qbs-borel*
shows (λ*x*. *qbs-nn-integral x f*) ∈ *all-meas-qbs X* $\to_Q$ *qbs-borel*
⟨proof⟩

lemma *qbs-nn-integral-return*:
assumes *f* ∈ *X* $\to_Q$ *qbs-borel*
and *x* ∈ *qbs-space X*
shows *qbs-nn-integral* (*return-qbs X x*) *f* = *f x*
⟨proof⟩

lemma *qbs-nn-integral-bind-all-meas*:
assumes [*qbs*]:*s* ∈ *qbs-space* (*all-meas-qbs X*)
f ∈ *X* $\to_Q$ *all-meas-qbs Y g* ∈ *Y* $\to_Q$ *qbs-borel*
shows *qbs-nn-integral* (*s* ⪢= *f*) *g* = *qbs-nn-integral s* (λ*y*. (*qbs-nn-integral* (*f y*) *g*)) (**is** *?lhs* = *?rhs*)
⟨proof⟩

lemmas *qbs-nn-integral-bind* = *qbs-nn-integral-bind-all-meas*[*OF monadM-all-meas-space qbs-morphism-monadAD*]

lemma *qbs-nn-integral-bind-return-all-meas*:
assumes [*qbs*]:*s* ∈ *qbs-space* (*all-meas-qbs Y*) *f* ∈ *Z* $\to_Q$ *qbs-borel g* ∈ *Y* $\to_Q$ *Z*
shows *qbs-nn-integral* (*s* ⪢= (λ*y*. *return-qbs Z* (*g y*))) *f* = *qbs-nn-integral s* (*f* ∘ *g*)
⟨proof⟩

lemmas *qbs-nn-integral-bind-return* = *qbs-nn-integral-bind-return-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-morphism*[*qbs*]:
qbs-integral ∈ *monadM-qbs X* $\to_Q$ (*X* $\Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ (*qbs-borel* :: (′*b* :: {*second-countable-topology*,*banach*}) *quasi-borel*)
⟨proof⟩

lemma *qbs-integral-morphism′*:
assumes [*qbs*,*measurable*]:*f* ∈ *X* $\to_Q$ *qbs-borel*
shows (λ*x*. *qbs-integral x f*) ∈ *all-meas-qbs X* $\to_Q$ *qbs-borel*
⟨proof⟩

lemma *qbs-integral-return*:
assumes [*qbs*]:*f* ∈ *X* $\to_Q$ *qbs-borel x* ∈ *qbs-space X*
shows *qbs-integral* (*return-qbs X x*) *f* = *f x*
⟨proof⟩

lemma
assumes [*qbs*]: *s* ∈ *qbs-space* (*all-meas-qbs X*) *f* ∈ *X* $\to_Q$ *all-meas-qbs Y g* ∈ *Y*

$\rightarrow_Q$ *qbs-borel*
and *qbs-integrable s* ($\lambda x.$ $\int_Q$ *y. norm* (*g y*) ∂f *x*) AE_Q *x in s. qbs-integrable* (*f x*) *g*
shows *qbs-integrable-bind-all-meas*: *qbs-integrable* (*s* $\gg=$ *f*) *g* (**is** *?goal1*)
and *qbs-integral-bind-all-meas*:($\int_Q$ *y. g y* ∂(*s* $\gg=$ *f*)) = ($\int_Q$ *x.* $\int_Q$ *y. g y* ∂f *x* ∂s) (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemmas *qbs-integrable-bind* = *qbs-integrable-bind-all-meas*[*OF monadM-all-meas-space qbs-morphism-monadAD*]
lemmas *qbs-integral-bind* = *qbs-integral-bind-all-meas*[*OF monadM-all-meas-space qbs-morphism-monadAD*]

lemma *qbs-integral-bind-return-all-meas*:
assumes [*qbs*]:*s* $\in$ *qbs-space* (*all-meas-qbs Y*) *f* $\in Z \rightarrow_Q$ *qbs-borel g* $\in Y \rightarrow_Q Z$
shows *qbs-integral* (*s* $\gg=$ ($\lambda y.$ *return-qbs Z* (*g y*))) *f* = *qbs-integral s* (*f* $\circ$ *g*)
⟨*proof*⟩

lemmas *qbs-integral-bind-return* = *qbs-integral-bind-return-all-meas*[*OF monadM-all-meas-space*]

4.1.13 Binary Product Measures

definition *qbs-pair-measure* :: [*'a qbs-measure*, *'b qbs-measure*] $\Rightarrow$ (*'a* $\times$ *'b*) *qbs-measure* (**infix** $\bigotimes_{Qmes}$ *80*) **where**
qbs-pair-measure-def':*qbs-pair-measure p q* $\equiv$ (*p* $\gg=$ ($\lambda x.$ *q* $\gg=$ ($\lambda y.$ *return-qbs* (*qbs-space-of p* $\bigotimes_Q$ *qbs-space-of q*) (*x*, *y*))))

context *pair-qbs-s-finites*
begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* $\times$ *real*) *measure*
⟨*proof*⟩

lemma
shows *qbs-pair-measure*: $[\![X, \alpha, \mu]\!]_{meas} \bigotimes_{Qmes} [\![Y, \beta, \nu]\!]_{meas}$ = $[\![X \bigotimes_Q Y$, *map-prod* α β $\circ$ *rr.from-real*, *distr* ($\mu \bigotimes_M \nu$) *borel rr.to-real*$]\!]_{meas}$
and *qbs-pair-measure-s-finite*: *qbs-s-finite* ($X \bigotimes_Q Y$) (*map-prod* α β $\circ$ *rr.from-real*) (*distr* ($\mu \bigotimes_M \nu$) *borel rr.to-real*)
⟨*proof*⟩

lemma *qbs-l-qbs-pair-measure*:
qbs-l ($[\![X, \alpha, \mu]\!]_{meas} \bigotimes_{Qmes} [\![Y, \beta, \nu]\!]_{meas}$) = *distr* ($\mu \bigotimes_M \nu$) (*qbs-to-measure* ($X \bigotimes_Q Y$)) (*map-prod* α β)
⟨*proof*⟩

lemma *qbs-nn-integral-pair-measure*:
assumes [*qbs*]:*f* $\in X \bigotimes_Q Y \rightarrow_Q$ *qbs-borel*
shows ($\int^+_Q$ *z. f z* ∂($[\![X, \alpha, \mu]\!]_{meas} \bigotimes_{Qmes} [\![Y, \beta, \nu]\!]_{meas}$)) = ($\int^+$ *z.* (*f* $\circ$

map-prod α β) *z* $\partial(\mu \bigotimes_M \nu)$)
⟨*proof*⟩

lemma *qbs-integral-pair-measure*:
assumes [*qbs*]:*f* $\in$ *X* $\bigotimes_Q$ *Y* $\to_Q$ *qbs-borel*
shows ($\int_Q$ *z*. *f z* ∂($[\![X, \alpha, \mu]\!]_{meas} \bigotimes_{Qmes} [\![Y, \beta, \nu]\!]_{meas}$)) = ($\int$ *z*. (*f* $\circ$ *map-prod* α β) *z* $\partial(\mu \bigotimes_M \nu)$)
⟨*proof*⟩

lemma *qbs-pair-measure-integrable-eq*:
fixes *f* :: - $\Rightarrow$ -::{*second-countable-topology*,*banach*}
shows *qbs-integrable* ($[\![X, \alpha, \mu]\!]_{meas} \bigotimes_{Qmes} [\![Y, \beta, \nu]\!]_{meas}$) *f* $\longleftrightarrow$ *f* $\in$ *X* $\bigotimes_Q$ *Y* $\to_Q$ *qbs-borel* $\wedge$ *integrable* ($\mu \bigotimes_M \nu$) (*f* $\circ$ (*map-prod* α β)) (**is** *?h* $\longleftrightarrow$ *?h1* $\wedge$ *?h2*)
⟨*proof*⟩

end

lemmas(**in** *pair-qbs-probs*) *qbs-pair-measure-prob* = *qbs-prob-axioms*

context
fixes *X Y p q*
assumes *p*[*qbs*]:*p* $\in$ *qbs-space* (*monadM-qbs X*) **and** *q*[*qbs*]:*q* $\in$ *qbs-space* (*monadM-qbs Y*)
begin

lemma *qbs-pair-measure-def*: *p* $\bigotimes_{Qmes}$ *q* = *p* $\gg\!=$ (λ*x*. *q* $\gg\!=$ (λ*y*. *return-qbs* (*X* $\bigotimes_Q$ *Y*) (*x*,*y*)))
⟨*proof*⟩

lemma *qbs-pair-measure-def2*: *p* $\bigotimes_{Qmes}$ *q* = *q* $\gg\!=$ (λ*y*. *p* $\gg\!=$ (λ*x*. *return-qbs* (*X* $\bigotimes_Q$ *Y*) (*x*,*y*)))
⟨*proof*⟩

lemma
assumes *f* $\in$ *X* $\bigotimes_Q$ *Y* $\to_Q$ *monadM-qbs Z*
shows *qbs-pair-bind-bind-return1'*:*q* $\gg\!=$ (λ*y*. *p* $\gg\!=$ (λ*x*. *f* (*x*,*y*))) = *p* $\bigotimes_{Qmes}$ *q* $\gg\!=$ *f*
and *qbs-pair-bind-bind-return2'*:*p* $\gg\!=$ (λ*x*. *q* $\gg\!=$ (λ*y*. *f* (*x*,*y*))) = *p* $\bigotimes_{Qmes}$ *q* $\gg\!=$ *f*
⟨*proof*⟩

lemma
assumes [*qbs*]:*f* $\in$ *X* $\to_Q$ *exp-qbs Y* (*monadM-qbs Z*)
shows *qbs-pair-bind-bind-return1''*: *q* $\gg\!=$ (λ*y*. *p* $\gg\!=$ (λ*x*. *f x y*)) = *p* $\bigotimes_{Qmes}$ *q* $\gg\!=$ (λ*x*. *f* (*fst x*) (*snd x*))
and *qbs-pair-bind-bind-return2''*: *p* $\gg\!=$ (λ*x*. *q* $\gg\!=$ (λ*y*. *f x y*)) = *p* $\bigotimes_{Qmes}$ *q* $\gg\!=$ (λ*x*. *f* (*fst x*) (*snd x*))
⟨*proof*⟩

lemma *qbs-nn-integral-Fubini-fst*:
assumes [*qbs*]:$f \in X \bigotimes_Q Y \rightarrow_Q$ *qbs-borel*
shows $(\int^+{}_Q\ x.\ \int^+{}_Q\ y.\ f\ (x,y)\ \partial q\ \partial p) = (\int^+{}_Q\ z.\ f\ z\ \partial(p \bigotimes_{Qmes} q))$
(**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-nn-integral-Fubini-snd*:
assumes [*qbs*]:$f \in X \bigotimes_Q Y \rightarrow_Q$ *qbs-borel*
shows $(\int^+{}_Q\ y.\ \int^+{}_Q\ x.\ f\ (x,y)\ \partial p\ \partial q) = (\int^+{}_Q\ z.\ f\ z\ \partial(p \bigotimes_{Qmes} q))$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma *qbs-ennintegral-indep-mult*:
assumes [*qbs*]: $f \in X \rightarrow_Q$ *qbs-borel* $g \in Y \rightarrow_Q$ *qbs-borel*
shows $(\int^+{}_Q\ z.\ f\ (fst\ z) * g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+{}_Q\ x.\ f\ x\ \partial p) * (\int^+{}_Q\ y.\ g\ y\ \partial q)$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

end

lemma *qbs-l-qbs-pair-measure*:
assumes *standard-borel M standard-borel N*
defines $X \equiv$ *measure-to-qbs M* **and** $Y \equiv$ *measure-to-qbs N*
assumes [*qbs*]: $p \in$ *qbs-space* (*monadM-qbs X*) $q \in$ *qbs-space* (*monadM-qbs Y*)
shows *qbs-l* $(p \bigotimes_{Qmes} q)$ = *qbs-l* $p \bigotimes_M$ *qbs-l* q
⟨*proof*⟩

lemma *qbs-pair-measure-morphism*[*qbs*]: *qbs-pair-measure* $\in$ *monadM-qbs X* $\rightarrow_Q$ *monadM-qbs Y* $\Rightarrow_Q$ *monadM-qbs* $(X \bigotimes_Q Y)$
⟨*proof*⟩

lemma *qbs-pair-measure-morphismP*: *qbs-pair-measure* $\in$ *monadP-qbs X* $\rightarrow_Q$ *monadP-qbs Y* $\Rightarrow_Q$ *monadP-qbs* $(X \bigotimes_Q Y)$
⟨*proof*⟩

lemma *qbs-nn-integral-indep1*:
assumes [*qbs*]:$p \in$ *qbs-space* (*monadM-qbs X*) $q \in$ *qbs-space* (*monadP-qbs X*) $f \in X \rightarrow_Q$ *qbs-borel*
shows $(\int^+{}_Q\ z.\ f\ (fst\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+{}_Q\ x.\ f\ x\ \partial p)$
⟨*proof*⟩

lemma *qbs-nn-integral-indep2*:
assumes [*qbs*]:$q \in$ *qbs-space* (*monadM-qbs Y*) $p \in$ *qbs-space* (*monadP-qbs X*) $f \in Y \rightarrow_Q$ *qbs-borel*
shows $(\int^+{}_Q\ z.\ f\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int^+{}_Q\ y.\ f\ y\ \partial q)$
⟨*proof*⟩

context

begin

interpretation *rr* : *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma *qbs-integrable-pair-swap*:
fixes *f* :: - ⇒ -::{*second-countable-topology*,*banach*}
assumes *p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadM-qbs Y*)
and *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) *f*
shows *qbs-integrable* (*q* $\bigotimes_{Qmes}$ *p*) (λ(*x*,*y*). *f* (*y*,*x*))
⟨*proof*⟩

lemma *qbs-integrable-pair1'*:
fixes *f* :: - ⇒ -::{*second-countable-topology*,*banach*}
assumes [*qbs*]:*p* ∈ *qbs-space* (*monadM-qbs X*)
q ∈ *qbs-space* (*monadM-qbs Y*)
f ∈ *X* $\bigotimes_Q$ *Y* $\to_Q$ *qbs-borel*
qbs-integrable p (λ*x*. $\int_Q$ *y*. *norm* (*f* (*x*,*y*)) ∂*q*)
and AE_Q *x in p*. *qbs-integrable q* (λ*y*. *f* (*x*,*y*))
shows *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) *f*
⟨*proof*⟩

lemma
assumes *p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadM-qbs Y*)
assumes *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) *f*
shows *qbs-integrable-pair1D1'*: *qbs-integrable p* (λ*x*. $\int_Q$ *y*. *f* (*x*,*y*) ∂*q*) (**is** *?g1*)
and *qbs-integrable-pair1D1-norm'*: *qbs-integrable p* (λ*x*. $\int_Q$ *y*. *norm* (*f* (*x*,*y*)) ∂*q*) (**is** *?g2*)
and *qbs-integrable-pair1D2'*: AE_Q *x in p*. *qbs-integrable q* (λ*y*. *f* (*x*,*y*)) (**is** *?g3*)
and *qbs-integrable-pair2D1'*: *qbs-integrable q* (λ*y*. $\int_Q$ *x*. *f* (*x*,*y*) ∂*p*) (**is** *?g4*)
and *qbs-integrable-pair2D1-norm'*: *qbs-integrable q* (λ*y*. $\int_Q$ *x*. *norm* (*f* (*x*,*y*)) ∂*p*) (**is** *?g5*)
and *qbs-integrable-pair2D2'*: AE_Q *y in q*. *qbs-integrable p* (λ*x*. *f* (*x*,*y*)) (**is** *?g6*)
and *qbs-integral-Fubini-fst'*: ($\int_Q$ *x*. $\int_Q$ *y*. *f* (*x*,*y*) ∂*q* ∂*p*) = ($\int_Q$ *z*. *f z* ∂(*p* $\bigotimes_{Qmes}$ *q*)) (**is** *?g7*)
and *qbs-integral-Fubini-snd'*: ($\int_Q$ *y*. $\int_Q$ *x*. *f* (*x*,*y*) ∂*p* ∂*q*) = ($\int_Q$ *z*. *f z* ∂(*p* $\bigotimes_{Qmes}$ *q*)) (**is** *?g8*)
⟨*proof*⟩

end

lemma
assumes *h*:*p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadM-qbs Y*) *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) (*case-prod f*)
shows *qbs-integrable-pair1D1*: *qbs-integrable p* (λ*x*. $\int_Q$ *y*. *f x y* ∂*q*)

and *qbs-integrable-pair1D1-norm*: *qbs-integrable p* (*λx.* $\int_Q$ *y. norm* (*f x y*) *∂q*)
and *qbs-integrable-pair1D2*: AE_Q *x in p. qbs-integrable q* (*λy. f x y*)
and *qbs-integrable-pair2D1*: *qbs-integrable q* (*λy.* $\int_Q$ *x. f x y ∂p*)
and *qbs-integrable-pair2D1-norm*: *qbs-integrable q* (*λy.* $\int_Q$ *x. norm* (*f x y*) *∂p*)
and *qbs-integrable-pair2D2*: AE_Q *y in q. qbs-integrable p* (*λx. f x y*)
and *qbs-integral-Fubini-fst*: ($\int_Q$ *x.* $\int_Q$ *y. f x y ∂q ∂p*) = ($\int_Q$ (*x,y*). *f x y ∂*(*p* $\bigotimes_{Qmes}$ *q*)) (**is** *?g7*)
and *qbs-integral-Fubini-snd*: ($\int_Q$ *y.* $\int_Q$ *x. f x y ∂p ∂q*) = ($\int_Q$ (*x,y*). *f x y ∂*(*p* $\bigotimes_{Qmes}$ *q*)) (**is** *?g8*)
⟨*proof*⟩

lemma *qbs-integrable-pair2'*:
fixes *f* :: - ⇒ -::{*second-countable-topology*,*banach*}
assumes *p* ∈ *qbs-space* (*monadM-qbs X*)
q ∈ *qbs-space* (*monadM-qbs Y*)
f ∈ *X* $\bigotimes_Q$ *Y* $\to_Q$ *qbs-borel*
qbs-integrable q (*λy.* $\int_Q$ *x. norm* (*f* (*x,y*)) *∂p*)
and AE_Q *y in q. qbs-integrable p* (*λx. f* (*x,y*))
shows *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) *f*
⟨*proof*⟩

lemma *qbs-integrable-indep-mult*:
fixes *f* :: - ⇒ -::{*real-normed-div-algebra*,*second-countable-topology*, *banach*}
assumes [*qbs*]:*p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadM-qbs Y*)
and *qbs-integrable p f qbs-integrable q g*
shows *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) (*λx. f* (*fst x*) ∗ *g* (*snd x*))
⟨*proof*⟩

lemma *qbs-integrable-indep1*:
fixes *f* :: - ⇒ -::{*real-normed-div-algebra*,*second-countable-topology*, *banach*}
assumes *p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadP-qbs Y*) *qbs-integrable p f*
shows *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) (*λx. f* (*fst x*))
⟨*proof*⟩

lemma *qbs-integral-indep1*:
fixes *f* :: - ⇒ -::{*real-normed-div-algebra*,*second-countable-topology*,*banach*}
assumes *p* ∈ *qbs-space* (*monadM-qbs X*) *q* ∈ *qbs-space* (*monadP-qbs Y*) *qbs-integrable p f*
shows ($\int_Q$ *z. f* (*fst z*) *∂*(*p* $\bigotimes_{Qmes}$ *q*)) = ($\int_Q$ *x. f x ∂p*)
⟨*proof*⟩

lemma *qbs-integrable-indep2*:
fixes *g* :: - ⇒ -::{*real-normed-div-algebra*,*second-countable-topology*, *banach*}
assumes *p* ∈ *qbs-space* (*monadP-qbs X*) *q* ∈ *qbs-space* (*monadM-qbs Y*) *qbs-integrable q g*
shows *qbs-integrable* (*p* $\bigotimes_{Qmes}$ *q*) (*λx. g* (*snd x*))
⟨*proof*⟩

lemma *qbs-integral-indep2*:
fixes g :: - ⇒ -::{*real-normed-div-algebra*,*second-countable-topology*}
assumes $p \in$ *qbs-space* (*monadP-qbs X*) $q \in$ *qbs-space* (*monadM-qbs Y*) *qbs-integrable q g*
shows $(\int_Q z.\ g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q y.\ g\ y\ \partial q)$
⟨*proof*⟩

lemma *qbs-integral-indep-mult1*:
fixes f **and** g:: - ⇒ -::{*real-normed-field*,*second-countable-topology*, *banach*}
assumes $p \in$ *qbs-space* (*monadP-qbs X*) $q \in$ *qbs-space* (*monadP-qbs Y*)
and *qbs-integrable p f qbs-integrable q g*
shows $(\int_Q z.\ f\ (fst\ z) * g\ (snd\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q x.\ f\ x\ \partial p) * (\int_Q y.\ g\ y\ \partial q)$
⟨*proof*⟩

lemma *qbs-integral-indep-mult2*:
fixes f **and** g:: - ⇒ -::{*real-normed-field*,*second-countable-topology*}
assumes $p \in$ *qbs-space* (*monadP-qbs X*) $q \in$ *qbs-space* (*monadP-qbs Y*)
and *qbs-integrable p f qbs-integrable q g*
shows $(\int_Q z.\ g\ (snd\ z) * f\ (fst\ z)\ \partial(p \bigotimes_{Qmes} q)) = (\int_Q y.\ g\ y\ \partial q) * (\int_Q x.\ f\ x\ \partial p)$
⟨*proof*⟩

4.1.14 The Inverse Function of *l*

definition *qbs-l-inverse* :: *'a measure* ⇒ *'a qbs-measure* **where**
qbs-l-inverse M ≡ $[\![$*measure-to-qbs M*, *from-real-into M*, *distr M borel* (*to-real-on M*)$]\!]_{meas}$

context *standard-borel-ne*
begin

lemma *qbs-l-inverse-def2*:
assumes [*measurable-cong*]: *sets* μ = *sets M*
shows *qbs-l-inverse* μ = $[\![$*measure-to-qbs M*, *from-real*, *distr* μ *borel to-real*$]\!]_{meas}$
⟨*proof*⟩

lemma
assumes [*measurable-cong*]:*sets* μ = *sets M*
shows *qbs-l-inverse-meas*: *qbs-meas* (*measure-to-qbs M*) *from-real* (*distr* μ *borel to-real*)
and *qbs-l-inverse-s-finite*: *s-finite-measure* μ ⟹ *qbs-s-finite* (*measure-to-qbs M*) *from-real* (*distr* μ *borel to-real*)
and *qbs-l-inverse-qbs-prob*: *prob-space* μ ⟹ *qbs-prob* (*measure-to-qbs M*) *from-real* (*distr* μ *borel to-real*)
⟨*proof*⟩

corollary
assumes *sets* μ = *sets M*

shows *qbs-l-inverse-in-space-all-meas*:*qbs-l-inverse* $\mu \in$ *qbs-space* (*all-meas-qbs* *M*)
and *qbs-l-inverse-in-space-monadM*: *s-finite-measure* $\mu \Longrightarrow$ *qbs-l-inverse* $\mu \in$ *qbs-space* (*monadM-qbs M*)
and *qbs-l-inverse-in-space-monadP*: *prob-space* $\mu \Longrightarrow$ *qbs-l-inverse* $\mu \in$ *qbs-space* (*monadP-qbs M*)
⟨*proof*⟩

lemma *qbs-l-qbs-l-inverse*:
assumes [*measurable-cong*]: *sets* μ = *sets M*
shows *qbs-l* (*qbs-l-inverse* μ) = μ
⟨*proof*⟩

lemma *qbs-l-inverse-qbs-l-all-meas*:
assumes $s \in$ *qbs-space* (*all-meas-qbs* (*measure-to-qbs M*))
shows *qbs-l-inverse* (*qbs-l s*) = *s*
⟨*proof*⟩

lemmas *qbs-l-inverse-qbs-l* = *qbs-l-inverse-qbs-l-all-meas*[*OF monadM-all-meas-space*]

lemmas *qbs-l-inverse-qbs-l-monadP* = *qbs-l-inverse-qbs-l*[*OF qbs-space-monadPM*]

lemma *qbs-l-inverse-morphism-kernel*:
assumes *measure-kernel N M k*
shows ($\lambda x.$ *qbs-l-inverse* (*k x*)) $\in$ *measure-to-qbs N* $\rightarrow_Q$ *all-meas-qbs* (*measure-to-qbs* *M*)
⟨*proof*⟩

lemma *qbs-l-inverse-morphism-s-finite*:
assumes *s-finite-kernel N M k*
shows ($\lambda x.$ *qbs-l-inverse* (*k x*)) $\in$ *measure-to-qbs N* $\rightarrow_Q$ *monadM-qbs* (*measure-to-qbs* *M*)
⟨*proof*⟩

lemma *qbs-l-inverse-qbs-morphism-prob*:
assumes [*measurable*]:$k \in N \rightarrow_M$ *prob-algebra M*
shows ($\lambda x.$ *qbs-l-inverse* (*k x*)) $\in$ *measure-to-qbs N* $\rightarrow_Q$ *monadP-qbs* (*measure-to-qbs* *M*)
⟨*proof*⟩

lemma *qbs-l-inverse-return*:
assumes $x \in$ *space M*
shows *qbs-l-inverse* (*return M x*) = *return-qbs* (*measure-to-qbs M*) *x*
⟨*proof*⟩

lemma *qbs-l-inverse-bind-kernel*:
assumes *standard-borel-ne N measure-kernel M N k*
shows *qbs-l-inverse* ($M \gg=_k k$) = *qbs-l-inverse M* $\gg=$ ($\lambda x.$ *qbs-l-inverse* (*k x*))
(**is** *?lhs* = *?rhs*)

⟨*proof*⟩

lemma *qbs-l-inverse-bind*:
  assumes *standard-borel-ne N s-finite-measure M k* ∈ *M* $\rightarrow_M$ *prob-algebra N*
  shows *qbs-l-inverse* (*M* ⤜ *k*) = *qbs-l-inverse M* ⤜ (λ*x*. *qbs-l-inverse* (*k x*))
  ⟨*proof*⟩

end

4.1.15 PMF and SPMF

definition *qbs-pmf* ≡ (λ*p*. *qbs-l-inverse* (*measure-pmf p*))
definition *qbs-spmf* ≡ (λ*p*. *qbs-l-inverse* (*measure-spmf p*))

declare [[*coercion qbs-pmf*]]

lemma *qbs-pmf-qbsP*:
  fixes *p* :: (- :: *countable*) *pmf*
  shows *qbs-pmf p* ∈ *qbs-space* (*monadP-qbs* (*count-space*$_Q$ *UNIV*))
  ⟨*proof*⟩

lemma *qbs-pmf-qbs*[*qbs*]:
  fixes *p* :: (- :: *countable*) *pmf*
  shows *qbs-pmf p* ∈ *qbs-space* (*monadM-qbs* (*count-space*$_Q$ *UNIV*))
  ⟨*proof*⟩

lemma *qbs-spmf-qbs*[*qbs*]:
  fixes *q* :: (- :: *countable*) *spmf*
  shows *qbs-spmf q* ∈ *qbs-space* (*monadM-qbs* (*count-space*$_Q$ *UNIV*))
  ⟨*proof*⟩

lemma [*simp*]:
  fixes *p* :: (- :: *countable*) *pmf* **and** *q* :: (- :: *countable*) *spmf*
  shows *qbs-l-qbs-pmf*: *qbs-l* (*qbs-pmf p*) = *measure-pmf p*
    and *qbs-l-qbs-spmf*: *qbs-l* (*qbs-spmf q*) = *measure-spmf q*
  ⟨*proof*⟩

lemma *qbs-pmf-return-pmf*:
  fixes *x* :: - :: *countable*
  shows *qbs-pmf* (*return-pmf x*) = *return-qbs* (*count-space*$_Q$ *UNIV*) *x*
⟨*proof*⟩

lemma *qbs-pmf-bind-pmf*:
  fixes *p* :: (′*a* :: *countable*) *pmf* **and** *f* :: ′*a* ⇒ (′*b* :: *countable*) *pmf*
  shows *qbs-pmf* (*p* ⤜ *f*) = *qbs-pmf p* ⤜ (λ*x*. *qbs-pmf* (*f x*))
  ⟨*proof*⟩

lemma *qbs-pair-pmf*:
  fixes *p* :: (′*a* :: *countable*) *pmf* **and** *q* :: (′*b* :: *countable*) *pmf*

shows *qbs-pmf p* $\bigotimes_{Qmes}$ *qbs-pmf q* = *qbs-pmf* (*pair-pmf p q*)
⟨*proof*⟩

4.1.16 Density

lift-definition *density-qbs* :: [′*a qbs-measure*, ′*a* ⇒ *ennreal*] ⇒ ′*a qbs-measure*
is $\lambda(X,\alpha,\mu)$ *f*. *if f* ∈ *X* $\to_Q$ *qbs-borel then* (*X*, α, *density* μ (*f* ∘ α)) *else* (*X*, *SOME a*. *a* ∈ *qbs-Mx X*, *null-measure borel*)
⟨*proof*⟩

lemma(**in** *qbs-meas*) *density-qbs*:
shows *f* ∈ *X* $\to_Q$ *qbs-borel* ⟹ *density-qbs* $[\![X,\alpha, \mu]\!]_{meas}$ *f* = $[\![X, \alpha,$ *density* μ (*f* ∘ α)$]\!]_{meas}$
⟨*proof*⟩

lemma (**in** *qbs-meas*) *density-qbs-meas*: *qbs-meas X* α (*density* μ (*f* ∘ α))
⟨*proof*⟩

lemma(**in** *qbs-s-finite*) *density-qbs-s-finite*:
f ∈ *X* $\to_Q$ *qbs-borel* ⟹ *qbs-s-finite X* α (*density* μ (*f* ∘ α))
⟨*proof*⟩

lemma *density-qbs-density-qbs-eq-all-meas*:
assumes [*qbs*]:*s* ∈ *qbs-space* (*all-meas-qbs X*) *f* ∈ *X* $\to_Q$ *qbs-borel g* ∈ *X* $\to_Q$ *qbs-borel*
shows *density-qbs* (*density-qbs s f*) *g* = *density-qbs s* (λ*x*. *f x* ∗ *g x*)
⟨*proof*⟩

lemmas *density-qbs-density-qbs-eq* = *density-qbs-density-qbs-eq-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-l-density-qbs-all-meas*:
assumes [*qbs*,*measurable*]:*s* ∈ *qbs-space* (*all-meas-qbs X*) *f* ∈ *X* $\to_Q$ *qbs-borel*
shows *qbs-l* (*density-qbs s f*) = *density* (*qbs-l s*) *f*
⟨*proof*⟩

lemmas *qbs-l-density-qbs* = *qbs-l-density-qbs-all-meas*[*OF monadM-all-meas-space*]

corollary *qbs-l-density-qbs-indicator-all-meas*:
assumes [*qbs*,*measurable*]:*s* ∈ *qbs-space* (*all-meas-qbs X*) *qbs-pred X P*
shows *qbs-l* (*density-qbs s* (*indicator* {*x*∈*qbs-space X*. *P x*})) (*qbs-space X*) = *qbs-l s* {*x*∈*qbs-space X*. *P x*}
⟨*proof*⟩

lemmas *qbs-l-density-qbs-indicator* = *qbs-l-density-qbs-indicator-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-nn-integral-density-qbs-all-meas*:
assumes [*qbs*,*measurable*]:*s* ∈ *qbs-space* (*all-meas-qbs X*) *f* ∈ *X* $\to_Q$ *qbs-borel g*

$\in X \to_Q$ *qbs-borel*
shows $(\int^{+}_{Q} x.\ g\ x\ \partial(\textit{density-qbs}\ s\ f)) = (\int^{+}_{Q} x.\ f\ x * g\ x\ \partial s)$
⟨*proof*⟩

lemmas *qbs-nn-integral-density-qbs* = *qbs-nn-integral-density-qbs-all-meas*[*OF monadM-all-meas-space*]

lemma *qbs-integral-density-qbs-all-meas*:
fixes $g :: 'a \Rightarrow 'b::\{banach,\ second\text{-}countable\text{-}topology\}$ **and** $f :: 'a \Rightarrow real$
assumes [*qbs,measurable*]:$s \in$ *qbs-space* (*all-meas-qbs* X) $f \in X \to_Q$ *qbs-borel* $g \in X \to_Q$ *qbs-borel*
and $AE_Q\ x\ in\ s.\ f\ x \geq 0$
shows $(\int_{Q} x.\ g\ x\ \partial(\textit{density-qbs}\ s\ f)) = (\int_{Q} x.\ f\ x *_R g\ x\ \partial s)$
⟨*proof*⟩

lemmas *qbs-integral-density-qbs* = *qbs-integral-density-qbs-all-meas*[*OF monadM-all-meas-space*]

lemma *density-qbs-morphism*[*qbs*]: *density-qbs* $\in$ *monadM-qbs* $X \to_Q (X \Rightarrow_Q$ *qbs-borel*$) \Rightarrow_Q$ *monadM-qbs* X
⟨*proof*⟩

lemma *density-qbs-morphism'*:
assumes [*qbs,measurable*]: $f \in X \Rightarrow_Q$ *qbs-borel*
shows ($\lambda p.$ *density-qbs* $p\ f$) $\in$ *all-meas-qbs* $X \Rightarrow_Q$ *all-meas-qbs* X
⟨*proof*⟩

lemma *density-qbs-cong-AE-all-meas*:
assumes [*qbs*]: $s \in$ *qbs-space* (*all-meas-qbs* X) $f \in X \to_Q$ *qbs-borel* $g \in X \to_Q$ *qbs-borel*
and $AE_Q\ x\ in\ s.\ f\ x = g\ x$
shows *density-qbs* $s\ f$ = *density-qbs* $s\ g$
⟨*proof*⟩

lemmas *density-qbs-cong-AE* = *density-qbs-cong-AE-all-meas*[*OF monadM-all-meas-space*]

corollary *density-qbs-cong-all-meas*:
assumes [*qbs*]: $s \in$ *qbs-space* (*all-meas-qbs* X) $f \in X \to_Q$ *qbs-borel*
and $\bigwedge x.\ x \in$ *qbs-space* $X \Longrightarrow f\ x = g\ x$
shows *density-qbs* $s\ f$ = *density-qbs* $s\ g$
⟨*proof*⟩

lemmas *density-qbs-cong* = *density-qbs-cong-all-meas*[*OF monadM-all-meas-space*]

lemma *density-qbs-1*[*simp*]: *density-qbs* s ($\lambda x.\ 1$) = s
⟨*proof*⟩

lemma *pair-density-qbs*:
assumes [*qbs*]: $p \in$ *qbs-space* (*monadM-qbs* X) $q \in$ *qbs-space* (*monadM-qbs* Y)
and [*qbs*]: $f \in X \to_Q$ *qbs-borel* $g \in Y \to_Q$ *qbs-borel*

shows *density-qbs p f* $\bigotimes_{Qmes}$ *density-qbs q g* = *density-qbs* (*p* $\bigotimes_{Qmes}$ *q*) (λ(*x*,*y*). *f x* ∗ *g y*)
⟨*proof*⟩

4.1.17 Normalization

definition *normalize-qbs* :: *'a qbs-measure* ⇒ *'a qbs-measure* **where**
normalize-qbs s ≡ (*let X* = *qbs-space-of s*;
r = *qbs-l s* (*qbs-space X*) *in*
if r ≠ *0* ∧ *r* ≠ ∞ *then density-qbs s* (λ*x*. *1* / *r*)
else qbs-null-measure X)

lemma
assumes *s* ∈ *qbs-space* (*all-meas-qbs X*)
shows *normalize-qbs-all-meas*: *qbs-l s* (*qbs-space X*) ≠ *0* ⟹ *qbs-l s* (*qbs-space X*) ≠ ∞ ⟹ *normalize-qbs s* = *density-qbs s* (λ*x*. *1* / *emeasure* (*qbs-l s*) (*qbs-space X*))
and *normalize-qbs0-all-meas*: *qbs-l s* (*qbs-space X*) = *0* ⟹ *normalize-qbs s* = *qbs-null-measure X*
and *normalize-qbsinfty-all-meas*: *qbs-l s* (*qbs-space X*) = ∞ ⟹ *normalize-qbs s* = *qbs-null-measure X*
⟨*proof*⟩

lemma
assumes *s* ∈ *qbs-space* (*monadM-qbs X*)
shows *normalize-qbs*: *qbs-l s* (*qbs-space X*) ≠ *0* ⟹ *qbs-l s* (*qbs-space X*) ≠ ∞ ⟹ *normalize-qbs s* = *density-qbs s* (λ*x*. *1* / *emeasure* (*qbs-l s*) (*qbs-space X*))
and *normalize-qbs0*: *qbs-l s* (*qbs-space X*) = *0* ⟹ *normalize-qbs s* = *qbs-null-measure X*
and *normalize-qbsinfty*: *qbs-l s* (*qbs-space X*) = ∞ ⟹ *normalize-qbs s* = *qbs-null-measure X*
⟨*proof*⟩

lemma *normalize-qbs-prob-all-meas*:
assumes *s* ∈ *qbs-space* (*all-meas-qbs X*) *qbs-l s* (*qbs-space X*) ≠ *0 qbs-l s* (*qbs-space X*) ≠ ∞
shows *normalize-qbs s* ∈ *qbs-space* (*monadP-qbs X*)
⟨*proof*⟩

lemmas *normalize-qbs-prob* = *normalize-qbs-prob-all-meas*[*OF monadM-all-meas-space*]

lemma *normalize-qbs-morphism*[*qbs*]: *normalize-qbs* ∈ *monadM-qbs X* $\rightarrow_Q$ *monadM-qbs X*
⟨*proof*⟩

lemma *normalize-qbs-morphismP*:
assumes [*qbs*]:*s* ∈ *X* $\rightarrow_Q$ *monadM-qbs Y*
and ⋀*x*. *x* ∈ *qbs-space X* ⟹ *qbs-l* (*s x*) (*qbs-space Y*) ≠ *0*
and ⋀*x*. *x* ∈ *qbs-space X* ⟹ *qbs-l* (*s x*) (*qbs-space Y*) ≠ ∞

shows $(\lambda x.\ normalize\text{-}qbs\ (s\ x)) \in X \to_Q monadP\text{-}qbs\ Y$
⟨*proof*⟩

lemma *normalize-qbs-monadP-ident*:
assumes $s \in qbs\text{-}space\ (monadP\text{-}qbs\ X)$
shows $normalize\text{-}qbs\ s = s$
⟨*proof*⟩

corollary *normalize-qbs-idenpotent*: $normalize\text{-}qbs\ (normalize\text{-}qbs\ s) = normalize\text{-}qbs\ s$
⟨*proof*⟩

4.1.18 Product Measures

definition $PiQ\text{-}measure :: [\prime a\ set, \prime a \Rightarrow \prime b\ qbs\text{-}measure] \Rightarrow (\prime a \Rightarrow \prime b)\ qbs\text{-}measure$
where
$PiQ\text{-}measure \equiv (\lambda I\ si.\ if\ (\forall i \in I.\ \exists Mi.\ standard\text{-}borel\text{-}ne\ Mi \wedge si\ i \in qbs\text{-}space\ (monadM\text{-}qbs\ (measure\text{-}to\text{-}qbs\ Mi)))$
$then\ if\ countable\ I \wedge (\forall i \in I.\ prob\text{-}space\ (qbs\text{-}l\ (si\ i)))\ then\ qbs\text{-}l\text{-}inverse\ (\Pi_M\ i \in I.\ qbs\text{-}l\ (si\ i))$
$else\ if\ finite\ I \wedge (\forall i \in I.\ sigma\text{-}finite\text{-}measure\ (qbs\text{-}l\ (si\ i)))\ then\ qbs\text{-}l\text{-}inverse\ (\Pi_M\ i \in I.\ qbs\text{-}l\ (si\ i))$
$else\ qbs\text{-}null\text{-}measure\ (\Pi_Q\ i \in I.\ qbs\text{-}space\text{-}of\ (si\ i))$
$else\ qbs\text{-}null\text{-}measure\ (\Pi_Q\ i \in I.\ qbs\text{-}space\text{-}of\ (si\ i)))$

syntax
-PiQ-measure :: *pttrn* ⇒ *'i set* ⇒ *'a qbs-measure* ⇒ (*'i* => *'a*) *qbs-measure* ((*3*Π_{Qmeas} -∈-./ -) *10*)
syntax-consts
-PiQ-measure ⇌ *PiQ-measure*
translations
Π_{Qmeas} *x*∈*I*. *X* == *CONST PiQ-measure I* (λ*x*. *X*)

context
fixes *I* **and** *Mi*
assumes *standard-borel-ne*:$\bigwedge i.\ i \in I \Longrightarrow standard\text{-}borel\text{-}ne\ (Mi\ i)$
begin

context
assumes *countableI*:*countable I*
begin

interpretation *sb*:*standard-borel-ne* Π_M *i*∈*I*. (*borel* :: *real measure*)
⟨*proof*⟩

interpretation *sbM*: *standard-borel-ne* Π_M *i*∈*I*. *Mi i*
⟨*proof*⟩

lemma

assumes $\bigwedge i.\ i \in I \Longrightarrow si\ i \in qbs\text{-}space\ (monadP\text{-}qbs\ (measure\text{-}to\text{-}qbs\ (Mi\ i)))$
and $\bigwedge i.\ i \in I \Longrightarrow si\ i = [\![measure\text{-}to\text{-}qbs\ (Mi\ i),\ \alpha\ i,\ \mu\ i]\!]_{meas}$ $\bigwedge i.\ i \in I \Longrightarrow qbs\text{-}prob\ (measure\text{-}to\text{-}qbs\ (Mi\ i))\ (\alpha\ i)\ (\mu\ i)$
shows *PiQ-measure-prob-eq*: $(\Pi_{Qmeas}\ i{\in}I.\ si\ i) = [\![measure\text{-}to\text{-}qbs\ (\Pi_M\ i{\in}I.\ Mi\ i),\ sbM.from\text{-}real,\ distr\ (\Pi_M\ i{\in}I.\ qbs\text{-}l\ (si\ i))\ borel\ sbM.to\text{-}real]\!]_{meas}$ (**is** - = *?rhs*)
and *PiQ-measure-qbs-prob*: $qbs\text{-}prob\ (measure\text{-}to\text{-}qbs\ (\Pi_M\ i{\in}I.\ Mi\ i))\ sbM.from\text{-}real\ (distr\ (\Pi_M\ i{\in}I.\ qbs\text{-}l\ (si\ i))\ borel\ sbM.to\text{-}real)$ (**is** *?qbsprob*)
⟨*proof*⟩

lemma *qbs-l-PiQ-measure-prob*:
assumes $\bigwedge i.\ i \in I \Longrightarrow si\ i \in qbs\text{-}space\ (monadP\text{-}qbs\ (measure\text{-}to\text{-}qbs\ (Mi\ i)))$
shows $qbs\text{-}l\ (\Pi_{Qmeas}\ i{\in}I.\ si\ i) = (\Pi_M\ i{\in}I.\ qbs\text{-}l\ (si\ i))$
⟨*proof*⟩

end

context
assumes *finI*: *finite I*
begin

interpretation *sb*:*standard-borel-ne* $\Pi_M\ i{\in}I.\ (borel :: real\ measure)$
⟨*proof*⟩

interpretation *sbM*: *standard-borel-ne* $\Pi_M\ i{\in}I.\ Mi\ i$
⟨*proof*⟩

lemma *qbs-l-PiQ-measure*:
assumes $\bigwedge i.\ i \in I \Longrightarrow si\ i \in qbs\text{-}space\ (monadM\text{-}qbs\ (measure\text{-}to\text{-}qbs\ (Mi\ i)))$
and $\bigwedge i.\ i \in I \Longrightarrow sigma\text{-}finite\text{-}measure\ (qbs\text{-}l\ (si\ i))$
shows $qbs\text{-}l\ (\Pi_{Qmeas}\ i{\in}I.\ si\ i) = (\Pi_M\ i{\in}I.\ qbs\text{-}l\ (si\ i))$
⟨*proof*⟩

end

end

4.2 Measures

4.2.1 The Lebesgue Measure

definition *lborel-qbs* ($lborel_Q$) **where** *lborel-qbs* ≡ *qbs-l-inverse lborel*

lemma *lborel-qbs-qbs*[*qbs*]: *lborel-qbs* ∈ *qbs-space* (*monadM-qbs qbs-borel*)
⟨*proof*⟩

lemma *qbs-l-lborel-qbs*[*simp*]: $qbs\text{-}l\ lborel_Q = lborel$
⟨*proof*⟩

corollary
shows *qbs-integral-lborel*: $(\int_Q x.\ f\ x\ \partial lborel\text{-}qbs) = (\int x.\ f\ x\ \partial lborel)$
and *qbs-nn-integral-lborel*: $(\int^+_Q x.\ f\ x\ \partial lborel\text{-}qbs) = (\int^+ x.\ f\ x\ \partial lborel)$
⟨*proof*⟩

lemma(**in** *standard-borel-ne*) *measure-with-args-morphism*:
assumes *s-finite-kernel X M k*
shows $qbs\text{-}l\text{-}inverse \circ k \in measure\text{-}to\text{-}qbs\ X \rightarrow_Q monadM\text{-}qbs\ (measure\text{-}to\text{-}qbs\ M)$
⟨*proof*⟩

lemma(**in** *standard-borel-ne*) *measure-with-args-morphismP*:
assumes [*measurable*]:$\mu \in X \rightarrow_M prob\text{-}algebra\ M$
shows $qbs\text{-}l\text{-}inverse \circ \mu \in measure\text{-}to\text{-}qbs\ X \rightarrow_Q monadP\text{-}qbs\ (measure\text{-}to\text{-}qbs\ M)$
⟨*proof*⟩

4.2.2 Counting Measure

abbreviation $counting\text{-}measure\text{-}qbs\ A \equiv qbs\text{-}l\text{-}inverse\ (count\text{-}space\ A)$

lemma *qbs-nn-integral-count-space-nat*:
fixes $f :: nat \Rightarrow ennreal$
shows $(\int^+_Q i.\ f\ i\ \partial counting\text{-}measure\text{-}qbs\ UNIV) = (\sum i.\ f\ i)$
⟨*proof*⟩

4.2.3 Normal Distribution

lemma *qbs-normal-distribution-qbs*: $(\lambda\mu\ \sigma.\ density\text{-}qbs\ lborel_Q\ (normal\text{-}density\ \mu\ \sigma)) \in qbs\text{-}borel \Rightarrow_Q qbs\text{-}borel \Rightarrow_Q monadM\text{-}qbs\ qbs\text{-}borel$
⟨*proof*⟩

lemma *qbs-l-qbs-normal-distribution*[*simp*]: $qbs\text{-}l\ (density\text{-}qbs\ lborel_Q\ (normal\text{-}density\ \mu\ \sigma)) = density\ lborel\ (normal\text{-}density\ \mu\ \sigma)$
⟨*proof*⟩

lemma *qbs-normal-distribution-P*: $\sigma > 0 \Longrightarrow density\text{-}qbs\ lborel_Q\ (normal\text{-}density\ \mu\ \sigma) \in qbs\text{-}space\ (monadP\text{-}qbs\ qbs\text{-}borel)$
⟨*proof*⟩

lemma *qbs-normal-distribution-integral*:
$(\int_Q x.\ f\ x\ \partial\ (density\text{-}qbs\ lborel_Q\ (normal\text{-}density\ \mu\ \sigma))) = (\int x.\ f\ x\ \partial\ (density\ lborel\ (\lambda x.\ ennreal\ (normal\text{-}density\ \mu\ \sigma\ x))))$
⟨*proof*⟩

lemma *qbs-normal-distribution-expectation*:
assumes [*measurable*]:$f \in borel\text{-}measurable\ borel$ **and** [*arith*]: $\sigma > 0$

shows ($\int_Q$ *x. f x* ∂ (*density-qbs* $lborel_Q$ (*normal-density* μ σ))) = ($\int$ *x. normal-density* μ σ *x* $*$ *f x* ∂ *lborel*)
⟨*proof*⟩

lemma *qbs-normal-posterior*:
assumes [*arith*]: $\sigma > 0$ $\sigma' > 0$
shows *normalize-qbs* (*density-qbs* (*density-qbs* $lborel_Q$ (*normal-density* μ σ)) (*normal-density* μ' σ')) = *density-qbs* $lborel_Q$ (*normal-density* (($\mu * \sigma'^2 + \mu' * \sigma^2$) / ($\sigma^2 + \sigma'^2$)) ($\sigma * \sigma'$ / *sqrt* ($\sigma^2 + \sigma'^2$))) (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

4.2.4 Uniform Distribution

definition *uniform-qbs* :: *'a qbs-measure* ⇒ *'a set* ⇒ *'a qbs-measure* **where**
uniform-qbs ≡ (λ*s A. qbs-l-inverse* (*uniform-measure* (*qbs-l s*) *A*))

lemma(**in** *standard-borel-ne*) *qbs-l-uniform-qbs'*:
assumes *sets* μ = *sets M* μ *A* $\neq$ *0*
shows *qbs-l* (*uniform-qbs* (*qbs-l-inverse* μ) *A*) = *uniform-measure* μ *A* (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

corollary(**in** *standard-borel-ne*) *qbs-l-uniform-qbs*:
assumes *s* ∈ *qbs-space* (*monadM-qbs* (*measure-to-qbs M*)) *qbs-l s A* $\neq$ *0*
shows *qbs-l* (*uniform-qbs s A*) = *uniform-measure* (*qbs-l s*) *A*
⟨*proof*⟩

lemma *interval-uniform-qbs*: (λ*a b. uniform-qbs* $lborel_Q$ {*a*<..<*b*::*real*}) ∈ $borel_Q$ $\Rightarrow_Q$ $borel_Q$ $\Rightarrow_Q$ *monadM-qbs* $borel_Q$
⟨*proof*⟩

context
fixes *a b* :: *real*
assumes [*arith*]:*a* < *b*
begin

lemma *qbs-uniform-distribution-expectation*:
assumes *f* ∈ *qbs-borel* $\to_Q$ *qbs-borel*
shows ($\int^+_Q$ *x. f x* ∂*uniform-qbs* $lborel_Q$ {*a*<..<*b*}) = ($\int^+$*x* ∈ {*a*<..<*b*}. *f x* ∂*lborel*) / (*b* − *a*)
⟨*proof*⟩

end

4.2.5 Bernoulli Distribution

abbreviation *qbs-bernoulli* :: *real* ⇒ *bool qbs-measure* **where**
qbs-bernoulli ≡ (λ*x. qbs-pmf* (*bernoulli-pmf x*))

lemma *bernoulli-measurable*:

$(\lambda x.\ measure\text{-}pmf\ (bernoulli\text{-}pmf\ x)) \in borel \rightarrow_M prob\text{-}algebra\ (count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *qbs-bernoulli-morphism*: $qbs\text{-}bernoulli \in qbs\text{-}borel \rightarrow_Q monadP\text{-}qbs\ (qbs\text{-}count\text{-}space\ UNIV)$
⟨*proof*⟩

lemma *qbs-bernoulli-expectation*:
assumes [*simp*]: $0 \leq p$ $p \leq 1$
shows $(\int_Q x.\ f\ x\ \partial qbs\text{-}bernoulli\ p) = f\ True * p + f\ False * (1 - p)$
⟨*proof*⟩

end

5 Examples

5.1 Montecarlo Approximation

theory *Montecarlo*
imports *Monad-QuasiBorel*
begin

declare [[*coercion qbs-l*]]

abbreviation *real-quasi-borel* :: *real quasi-borel* ($\mathbb{R}_Q$) **where**
real-quasi-borel ≡ *qbs-borel*
abbreviation *nat-quasi-borel* :: *nat quasi-borel* ($\mathbb{N}_Q$) **where**
nat-quasi-borel ≡ *qbs-count-space UNIV*

primrec *montecarlo* :: *'a qbs-measure* ⇒ (*'a* ⇒ *real*) ⇒ *nat* ⇒ *real qbs-measure*
where
montecarlo - - 0 = *return-qbs* $\mathbb{R}_Q$ *0* |
montecarlo d h (*Suc n*) = *do* { *m* ← *montecarlo d h n*;
x ← *d*;
return-qbs $\mathbb{R}_Q$ ((*h x* + *m* ∗ *real n*) / *real* (*Suc n*))}

declare
bind-qbs-morphismP[*qbs*]
return-qbs-morphismP[*qbs*]
qbs-pair-measure-morphismP[*qbs*]
qbs-space-monadPM[*qbs*]

lemma *montecarlo-qbs-morphism*[*qbs*]: $montecarlo \in qbs\text{-}space\ (monadP\text{-}qbs\ X \Rightarrow_Q (X \Rightarrow_Q \mathbb{R}_Q) \Rightarrow_Q \mathbb{N}_Q \Rightarrow_Q monadP\text{-}qbs\ \mathbb{R}_Q)$
⟨*proof*⟩

lemma *qbs-integrable-indep-mult2*:
fixes *f* :: - ⇒ *real*

assumes $p \in$ *qbs-space* (*monadM-qbs* X) $q \in$ *qbs-space* (*monadM-qbs* Y)
and *qbs-integrable p f*
and *qbs-integrable q g*
shows *qbs-integrable* ($p \bigotimes_{Qmes} q$) ($\lambda x.$ g (*snd* x) $*$ f (*fst* x))
⟨*proof*⟩

lemma *montecarlo-integrable*:
assumes [*qbs*]:$p \in$ *qbs-space* (*monadP-qbs* X) $h \in X \rightarrow_Q \mathbb{R}_Q$ *qbs-integrable p h qbs-integrable p* ($\lambda x.$ $h\ x * h\ x$)
shows *qbs-integrable* (*montecarlo p h n*) ($\lambda x.$ x) *qbs-integrable* (*montecarlo p h n*) ($\lambda x.$ $x * x$)
⟨*proof*⟩

lemma
fixes *n* :: *nat*
assumes [*qbs,simp*]:$p \in$ *qbs-space* (*monadP-qbs* X) $h \in X \rightarrow_Q \mathbb{R}_Q$ *qbs-integrable p h qbs-integrable p* ($\lambda x.$ $h\ x * h\ x$)
and e:$e > 0$
and ($\int_Q x.\ h\ x\ \partial p$) $= \mu$ ($\int_Q x.\ (h\ x - \mu)^2\ \partial p$) $= \sigma^2$
and n:$n > 0$
shows $\mathcal{P}$(y *in montecarlo p h n*. $|y - \mu| \geq e$) $\leq \sigma^2$ / (*real* $n * e^2$) (**is** *?P* $\leq$ -)
⟨*proof*⟩

end

5.2 Query

theory *Query*
imports *Monad-QuasiBorel*
begin

declare [[*coercion qbs-l*]]
abbreviation *qbs-real* :: *real quasi-borel* ($\mathbb{R}_Q$) **where** $\mathbb{R}_Q \equiv$ *qbs-borel*
abbreviation *qbs-ennreal* :: *ennreal quasi-borel* ($\mathbb{R}_{Q\geq 0}$) **where** $\mathbb{R}_{Q\geq 0} \equiv$ *qbs-borel*
abbreviation *qbs-nat* :: *nat quasi-borel* ($\mathbb{N}_Q$) **where** $\mathbb{N}_Q \equiv$ *qbs-count-space UNIV*
abbreviation *qbs-bool* :: *bool quasi-borel* ($\mathbb{B}_Q$) **where** $\mathbb{B}_Q \equiv$ *count-space*$_Q$ *UNIV*

definition *query* :: [*'a qbs-measure*, *'a* $\Rightarrow$ *ennreal*] $\Rightarrow$ *'a qbs-measure* **where**
query $\equiv$ ($\lambda s\ f.$ *normalize-qbs* (*density-qbs s f*))

lemma *query-qbs-morphism*[*qbs*]: *query* $\in$ *monadM-qbs* $X \rightarrow_Q$ ($X \Rightarrow_Q$ *qbs-borel*) $\Rightarrow_Q$ *monadM-qbs* X
⟨*proof*⟩

definition *condition* $\equiv$ ($\lambda s\ P.$ *query s* ($\lambda x.$ *if P x then 1 else 0*))

lemma *condition-qbs-morphism*[*qbs*]: *condition* $\in$ *monadM-qbs X* $\Rightarrow_Q$ ($X \Rightarrow_Q \mathbb{B}_Q$) $\Rightarrow_Q$ *monadM-qbs X*
⟨*proof*⟩

lemma *condition-morphismP*:
assumes $\bigwedge x.\ x \in$ *qbs-space X* $\Longrightarrow \mathcal{P}$(*y in qbs-l* (*s x*). *P x y*) $\neq$ *0*
and [*qbs*]: $s \in X \rightarrow_Q$ *monadP-qbs Y P* $\in X \rightarrow_Q Y \Rightarrow_Q$ *qbs-count-space UNIV*
shows ($\lambda x.$ *condition* (*s x*) (*P x*)) $\in X \rightarrow_Q$ *monadP-qbs Y*
⟨*proof*⟩

lemma *query-Bayes*:
assumes [*qbs*]: $s \in$ *qbs-space* (*monadP-qbs X*) *qbs-pred X P qbs-pred X Q*
shows $\mathcal{P}$(*x in condition s P. Q x*) = $\mathcal{P}$(*x in s. Q x* | *P x*) (**is** *?lhs* = *?pq*)
⟨*proof*⟩

lemma *qbs-pmf-cond-pmf*:
fixes *p* :: *'a* :: *countable pmf*
assumes *set-pmf p* $\cap$ {*x. P x*} $\neq$ {}
shows *condition* (*qbs-pmf p*) *P* = *qbs-pmf* (*cond-pmf p* {*x. P x*})
⟨*proof*⟩

5.2.1 `twoUs`

Example from Section 2 in [3].

definition *Uniform* $\equiv$ (λ*a b*::*real. uniform-qbs lborel-qbs* {*a*<..<*b*})

lemma *Uniform-qbs*[*qbs*]: *Uniform* $\in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q$
⟨*proof*⟩

definition *twoUs* :: (*real* × *real*) *qbs-measure* **where**
twoUs $\equiv$ *do* {
let u1 = *Uniform 0 1*;
let u2 = *Uniform 0 1*;
let y = *u1* $\bigotimes_{Qmes}$ *u2*;
condition y (λ(*x*,*y*). *x* < *0.5* $\vee$ *y* > *0.5*)
}

lemma *twoUs-qbs*: *twoUs* $\in$ *monadM-qbs* ($\mathbb{R}_Q \bigotimes_Q \mathbb{R}_Q$)
⟨*proof*⟩

interpretation *rr*: *standard-borel-ne borel* $\bigotimes_M$ *borel* :: (*real* × *real*) *measure*
⟨*proof*⟩

lemma *qbs-l-Uniform*[*simp*]: *a* < *b* $\Longrightarrow$ *qbs-l* (*Uniform a b*) = *uniform-measure lborel* {*a*<..<*b*}
⟨*proof*⟩

lemma *Uniform-qbsP*:
assumes [*arith*]: *a* < *b*

shows *Uniform a b* $\in$ *monadP-qbs* $\mathbb{R}_Q$
⟨*proof*⟩

interpretation *UniformP-pair*: *pair-prob-space uniform-measure lborel* {*0<..<1*::*real*} *uniform-measure lborel* {*0<..<1*::*real*}
⟨*proof*⟩

lemma *qbs-l-Uniform-pair*: $a < b \Longrightarrow$ *qbs-l* (*Uniform a b* $\bigotimes_{Qmes}$ *Uniform a b*) = *uniform-measure lborel* {*a<..<b*} $\bigotimes_M$ *uniform-measure lborel* {*a<..<b*}
⟨*proof*⟩

lemma *Uniform-pair-qbs*[*qbs*]:
assumes $a < b$
shows *Uniform a b* $\bigotimes_{Qmes}$ *Uniform a b* $\in$ *qbs-space* (*monadP-qbs* ($\mathbb{R}_Q$ $\bigotimes_Q$ $\mathbb{R}_Q$))
⟨*proof*⟩

lemma *twoUs-prob1*: $\mathcal{P}$(*z in Uniform 0 1* $\bigotimes_{Qmes}$ *Uniform 0 1*. *fst z* $< 0.5 \vee$ *snd z* > 0.5) $= 3 / 4$
⟨*proof*⟩

lemma *twoUs-prob2*:$\mathcal{P}$(*z in Uniform 0 1* $\bigotimes_{Qmes}$ *Uniform 0 1*. $1/2 <$ *fst z* $\wedge$ (*fst z* $< 1/2 \vee$ *snd z* $> 1/2$)) $= 1 / 4$
⟨*proof*⟩

lemma *twoUs-qbs-prob*: *twoUs* $\in$ *qbs-space* (*monadP-qbs* ($\mathbb{R}_Q$ $\bigotimes_Q$ $\mathbb{R}_Q$))
⟨*proof*⟩

lemma $\mathcal{P}$((*x*,*y*) *in twoUs*. $1/2 < x$) $= 1 / 3$
⟨*proof*⟩

5.2.2 Two Dice

Example from Adrian [2, Sect. 2.3].

abbreviation *die* $\equiv$ *qbs-pmf* (*pmf-of-set* {*Suc 0..6*})

lemma *die-qbs*[*qbs*]: *die* $\in$ *monadM-qbs* $\mathbb{N}_Q$
⟨*proof*⟩

definition *two-dice* :: *nat qbs-measure* **where**
two-dice $\equiv$ *do* {
  let die1 = die;
  let die2 = die;
  let twodice = die1 $\bigotimes_{Qmes}$ *die2*;
  (*x*,*y*) $\leftarrow$ *condition twodice*
    (λ(*x*,*y*). $x = 4 \vee y = 4$);
  return-qbs $\mathbb{N}_Q$ ($x + y$)
}

lemma *two-dice-qbs*: *two-dice* $\in$ *monadM-qbs* $\mathbb{N}_Q$
⟨*proof*⟩

lemma *prob-die2*: $\mathcal{P}$(*x in qbs-l* (*die* $\bigotimes_{Qmes}$ *die*). *P x*) = *real* (*card* ({*x*. *P x*} ∩ ({*1..6*} × {*1..6*}))) / *36* (**is** *?P* = *?rhs*)
⟨*proof*⟩

lemma *dice-prob1*: $\mathcal{P}$(*z in qbs-l* (*die* $\bigotimes_{Qmes}$ *die*). *fst z* = *4* ∨ *snd z* = *4*) = *11* / *36*
⟨*proof*⟩

lemma *dice-program-prob*:$\mathcal{P}$(*x in two-dice*. *P x*) = *2* ∗ ($\sum$ *n*∈{*5*,*6*,*7*,*9*,*10*}. *of-bool* (*P n*) / *11*) + *of-bool* (*P 8*) / *11* (**is** *?P* = *?rp*)
⟨*proof*⟩

corollary
$\mathcal{P}$(*x in two-dice*. *x* = *5*) = *2* / *11*
$\mathcal{P}$(*x in two-dice*. *x* = *6*) = *2* / *11*
$\mathcal{P}$(*x in two-dice*. *x* = *7*) = *2* / *11*
$\mathcal{P}$(*x in two-dice*. *x* = *8*) = *1* / *11*
$\mathcal{P}$(*x in two-dice*. *x* = *9*) = *2* / *11*
$\mathcal{P}$(*x in two-dice*. *x* = *10*) = *2* / *11*

⟨*proof*⟩

5.2.3 Gaussian Mean Learning

Example from Sato et al. Section 8. 2 in [3].

definition *Gauss* ≡ (λμ σ. *density-qbs lborel*$_Q$ (*normal-density* μ σ))

lemma *Gauss-qbs*[*qbs*]: *Gauss* $\in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q$
⟨*proof*⟩

primrec *GaussLearn′* :: [*real*, *real qbs-measure*, *real list*]
⇒ *real qbs-measure* **where**
GaussLearn′ - *p* [] = *p*
| *GaussLearn′* σ *p* (*y*#*ls*) = *query* (*GaussLearn′* σ *p ls*)
(*normal-density y* σ)

lemma *GaussLearn′-qbs*[*qbs*]:*GaussLearn′* $\in \mathbb{R}_Q \Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q \Rightarrow_Q$ *list-qbs* $\mathbb{R}_Q \Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q$
⟨*proof*⟩

context
fixes σ :: *real*
assumes [*arith*]: σ > *0*
begin

abbreviation *GaussLearn* ≡ *GaussLearn′ σ*

lemma *GaussLearn-qbs*[*qbs*]: *GaussLearn* ∈ *qbs-space* (*monadM-qbs* $\mathbb{R}_Q$ $\Rightarrow_Q$ *list-qbs* $\mathbb{R}_Q$ $\Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q$)
⟨*proof*⟩

definition *Total* :: *real list* ⇒ *real* **where** *Total* = (*λl. foldr* (+) *l 0*)

lemma *Total-simp*: *Total* [] = *0 Total* (*y#ls*) = *y* + *Total ls*
⟨*proof*⟩

lemma *Total-qbs*[*qbs*]: *Total* ∈ *list-qbs* $\mathbb{R}_Q$ $\to_Q$ $\mathbb{R}_Q$
⟨*proof*⟩

lemma *GaussLearn-Total*:
assumes [*arith*]: *ξ* > *0 n* = *length L*
shows *GaussLearn* (*Gauss δ ξ*) *L* = *Gauss* ((*Total L*$*\xi^2$+$\delta*\sigma^2$)/($n*\xi^2$+σ^2)) (*sqrt* (($\xi^2*\sigma^2$)/($n*\xi^2$+σ^2)))
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem1*:
fixes *a* :: *real*
assumes [*arith*]: *a* > *0 b* > *0 c* > *0 d* > *0*
shows (*λn. ln* ((*b* ∗ (*n* ∗ *d* + *c*)) / (*d* ∗ (*n* ∗ *b* + *a*)))) ⟶ *0*
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem1′*:
fixes *b* :: *real*
assumes [*arith*]: *b* > *0 d* > *0 s* > *0*
shows (*λn. ln* (*sqrt* (b^2 ∗ s^2 / (*real n* ∗ b^2 + s^2)) / *sqrt* (d^2 ∗ s^2 / (*real n* ∗ d^2 + s^2)))) ⟶ *0* (**is** *?f* ⟶ *0*)
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem2*:
fixes *s* :: *real*
assumes [*arith*]: *s* > *0 b* > *0 d* > *0*
shows (*λn.* ((*d* ∗ *s*) / (*n* ∗ *d* + *s*)) / (*2* ∗ ((*b* ∗ *s*) / (*n* ∗ *b* + *s*)))) ⟶ *1* / *2*
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem2′*:
fixes *s* :: *real*
assumes [*arith*]: *s* > *0 b* > *0 d* > *0*
shows (*λn.* ((*d*^*2* ∗ *s*^*2*) / (*n* ∗ *d*^*2* + *s*^*2*)) / (*2* ∗ ((*b*^*2* ∗ *s*^*2*) / (*n* ∗ *b*^*2* + *s*^*2*))) − *1* / *2*) ⟶ *0*
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem3*:
fixes *a b c d s K L* :: *real*
assumes [*arith*]: *b* > *0 d* > *0 s* > *0*

shows $((K * d + c * s) \ / \ (n * d + s) - (L * b + a * s) \ / \ (n * b + s))\hat{\ }2 \ / \ (2 * ((b * s) \ / \ (n * b + s))) = ((((((K - L) * d * b * real\ n + c * s * b * real\ n + K * d * s + c * s * s) - a * s * d * real\ n - L * b * s - a * s * s))^2 \ / \ (d * d * b * (real\ n * real\ n * real\ n) + s * s * b * real\ n + 2 * d * s * b * (real\ n * real\ n) + d * d * (real\ n * real\ n) * s + s * s * s + 2 * d * s * s * real\ n))) \ / \ (2 * (b * s))$ (**is** $?lhs = ?rhs$)
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem4*:
fixes $a\ b\ c\ d\ s\ K\ L :: real$
assumes $[arith]$: $b > 0\ d > 0\ s > 0$
shows $(\lambda n.\ (|c * s * b * real\ n| + |K * (real\ n) * d * s| + |c * s * s| + |a * s * d * real\ n| + |L * (real\ n) * b * s| + |a * s * s|)^2 \ / \ (d * d * b * (real\ n * real\ n * real\ n) + s * s * b * real\ n + 2 * d * s * b * (real\ n * real\ n) + d * d * (real\ n * real\ n) * s + s * s * s + 2 * d * s * s * real\ n) \ / \ (2 * (b * s))) \longrightarrow 0$ (**is** $(\lambda n.\ ?f\ n) \longrightarrow 0$)
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem5*:
fixes $a\ b\ c\ d\ K :: real$
assumes $[arith]$: $b > 0\ d > 0\ s > 0\ K > 0\ |f\ l| < K * length\ l$
shows $|(c * s * b * real\ (length\ l) + f\ l * d * s + c * s * s - a * s * d * real\ (length\ l) - f\ l * b * s - a * s * s)^2 \ / \ (d * d * b * (real\ (length\ l) * real\ (length\ l) * real\ (length\ l)) + s * s * b * real\ (length\ l) + 2 * d * s * b * (real\ (length\ l) * real\ (length\ l)) + d * d * (real\ (length\ l) * real\ (length\ l)) * s + s * s * s + 2 * d * s * s * real\ (length\ l)) \ / \ (2 * (b * s))| \leq |(|c * s * b * real\ (length\ l)| + |K * real\ (length\ l) * d * s| + |c * s * s| + |a * s * d * real\ (length\ l)| + |- K * real\ (length\ l) * b * s| + |a * s * s|)^2 \ / \ (d * d * b * (real\ (length\ l) * real\ (length\ l) * real\ (length\ l)) + s * s * b * real\ (length\ l) + 2 * d * s * b * (real\ (length\ l) * real\ (length\ l)) + d * d * (real\ (length\ l) * real\ (length\ l)) * s + s * s * s + 2 * d * s * s * real\ (length\ l)) \ / \ (2 * (b * s))|$ (**is** $|(?l)\hat{\ }2 \ / \ ?c1 \ / \ ?c2| \leq |(?r)\hat{\ }2 \ / \ - \ / \ -|$)
⟨*proof*⟩

lemma *GaussLearn-KL-divergence-lem6*:
fixes $a\ e\ b\ c\ d\ K :: real$ **and** $f :: 'a\ list \Rightarrow real$
assumes $[arith]$:$e > 0\ b > 0\ d > 0\ s > 0$
shows $\exists N.\ \forall l.\ length\ l \geq N \longrightarrow |f\ l| < K * length\ l \longrightarrow |((f\ l * d + c * s) \ / \ (length\ l * d + s) - (f\ l * b + a * s) \ / \ (length\ l * b + s))\hat{\ }2 \ / \ (2 * ((b * s) \ / \ (length\ l * b + s)))\ | < e$
⟨*proof*⟩

lemma *GaussLearn-KL-divergence*:
fixes $a\ b\ c\ d\ e\ K :: real$
assumes $[arith]$:$e > 0\ b > 0\ d > 0$
shows $\exists N.\ \forall L.\ length\ L > N \longrightarrow |Total\ L \ / \ length\ L| < K$
$\longrightarrow KL\text{-}divergence\ (exp\ 1)\ (GaussLearn\ (Gauss\ a\ b)\ L)\ (GaussLearn\ (Gauss\ c\ d)\ L) < e$
⟨*proof*⟩

end

5.2.4 Continuous Distributions

The following (highr-order) program receives a non-negative function f and returns the distribution whose density function is (noramlized) f if f is integrable w.r.t. the Lebesgue measure.

definition *dens-to-dist* :: [*'a* :: *euclidean-space* ⇒ *real*] ⇒ *'a qbs-measure* **where**
dens-to-dist ≡ (λ*f*. *do* {
    query lborel$_Q$ *f*
  })

lemma *dens-to-dist-qbs*[*qbs*]: *dens-to-dist* ∈ (*borel*$_Q$ ⇒$_Q$ ℝ$_Q$) →$_Q$ *monadM-qbs borel*$_Q$
⟨*proof*⟩

context
fixes *f* :: *'a* :: *euclidean-space* ⇒ *real*
assumes *f-qbs*[*qbs*]: *f* ∈ *qbs-borel* →$_Q$ ℝ$_Q$
and *f-le0*:⋀*x*. *f x* ≥ *0*
and *f-int-ne0*:*qbs-l* (*density-qbs lborel-qbs f*) *UNIV* ≠ *0*
and *f-integrable*: *qbs-integrable lborel-qbs f*
begin

lemma *f-integrable'*[*measurable*]: *integrable lborel f*
⟨*proof*⟩

lemma *f-int-neinfty*:
qbs-l (*density-qbs lborel-qbs f*) *UNIV* ≠ ∞
⟨*proof*⟩

lemma *dens-to-dist*: *dens-to-dist f* = *density-qbs lborel-qbs* (λ*x*. *ennreal* (*1* / *measure* (*qbs-l* (*density-qbs lborel-qbs f*)) *UNIV* ∗ *f x*))
⟨*proof*⟩

corollary *qbs-l-dens-to-dist*: *qbs-l* (*dens-to-dist f*) = *density lborel* (λ*x*. *ennreal* (*1* / *measure* (*qbs-l* (*density-qbs lborel-qbs f*)) *UNIV* ∗ *f x*))
⟨*proof*⟩

corollary *qbs-integral-dens-to-dist*:
assumes [*qbs*]: *g* ∈ *qbs-borel* →$_Q$ ℝ$_Q$
shows (∫$_Q$ *x*. *g x* ∂*dens-to-dist f*) = (∫$_Q$ *x*. *1* / *measure* (*qbs-l* (*density-qbs lborel-qbs f*)) *UNIV* ∗ *f x* ∗ *g x* ∂*lborel*$_Q$)
⟨*proof*⟩

lemma *dens-to-dist-prob*[*qbs*]:*dens-to-dist f* ∈ *qbs-space* (*monadP-qbs borel*$_Q$)
⟨*proof*⟩

end

5.2.5 Normal Distribution

context
  fixes μ σ :: *real*
  assumes *sigma-pos*[*arith*]: $\sigma > 0$
begin

We use an unnormalized density function.

definition *normal-f* $\equiv$ ($\lambda x.$ *exp* $(-(x - \mu)^2/ (2 * \sigma^2))$)

lemma *nc-normal-f*: *qbs-l* (*density-qbs lborel-qbs normal-f*) *UNIV* = *ennreal* (*sqrt* $(2 * pi * \sigma^2)$)
⟨*proof*⟩

corollary *measure-qbs-l-dens-to-dist-normal-f*: *measure* (*qbs-l* (*density-qbs lborel-qbs normal-f*)) *UNIV* = *sqrt* $(2 * pi * \sigma^2)$
  ⟨*proof*⟩

lemma *normal-f*:
  shows *normal-f* $\in$ *qbs-borel* $\to_Q$ $\mathbb{R}_Q$
    and $\bigwedge x.$ *normal-f* $x \geq 0$
    and *qbs-l* (*density-qbs lborel-qbs normal-f*) *UNIV* $\neq 0$
    and *qbs-integrable lborel-qbs normal-f*
  ⟨*proof*⟩

lemma *qbs-l-densto-dist-normal-f*: *qbs-l* (*dens-to-dist normal-f*) = *density lborel* (*normal-density* μ σ)
  ⟨*proof*⟩

end

5.2.6 Half Normal Distribution

context
  fixes μ σ :: *real*
  assumes *sigma-pos*[*arith*]:$\sigma > 0$
begin

definition *hnormal-f* $\equiv$ ($\lambda x.$ *if* $x \leq \mu$ *then 0 else normal-density* μ σ x)

lemma *nc-hnormal-f*: *qbs-l* (*density-qbs lborel-qbs hnormal-f*) *UNIV* = *ennreal* $(1/ 2)$
⟨*proof*⟩

corollary *measure-qbs-l-dens-to-dist-hnormal-f*: *measure* (*qbs-l* (*density-qbs lborel-qbs hnormal-f*)) *UNIV* = $1 / 2$
  ⟨*proof*⟩

lemma *hnormal-f*:

shows *hnormal-f* ∈ *qbs-borel* $\rightarrow_Q$ $\mathbb{R}_Q$
and ⋀*x*. *hnormal-f x* ≥ *0*
and *qbs-l* (*density-qbs lborel-qbs hnormal-f*) *UNIV* ≠ *0*
and *qbs-integrable lborel-qbs hnormal-f*
⟨*proof*⟩

lemma *qbs-l* (*dens-to-dist local.hnormal-f*) = *density lborel* (λ*x*. *ennreal* (*2* ∗ (*if x* ≤ μ *then 0 else normal-density* μ σ *x*)))
⟨*proof*⟩

end

5.2.7 Erlang Distribution

context
fixes *k* :: *nat* **and** *l* :: *real*
assumes *l-pos*[*arith*]: *l* > *0*
begin

definition *erlang-f* ≡ (λ*x*. *if x* < *0 then 0 else x*^*k* ∗ *exp* (− *l* ∗ *x*))

lemma *nc-erlang-f*: *qbs-l* (*density-qbs lborel-qbs erlang-f*) *UNIV* = *ennreal* (*fact k* / *l*^(*Suc k*))
⟨*proof*⟩

corollary *measure-qbs-l-dens-to-dist-erlang-f*: *measure* (*qbs-l* (*density-qbs lborel-qbs erlang-f*)) *UNIV* = *fact k* / *l*^(*Suc k*)
⟨*proof*⟩

lemma *erlang-f*:
shows *erlang-f* ∈ *qbs-borel* $\rightarrow_Q$ $\mathbb{R}_Q$
and ⋀*x*. *erlang-f x* ≥ *0*
and *qbs-l* (*density-qbs lborel-qbs erlang-f*) *UNIV* ≠ *0*
and *qbs-integrable lborel-qbs erlang-f*
⟨*proof*⟩

lemma *qbs-l* (*dens-to-dist erlang-f*) = *density lborel* (*erlang-density k l*)
⟨*proof*⟩

end

5.2.8 Uniform Distribution on $(0, 1) \times (0, 1)$.

definition *uniform-f* ≡ *indicat-real* ({*0*<..<*1*::*real*}×{*0*<..<*1*::*real*})

lemma
shows *uniform-f-qbs'*[*qbs*]: *uniform-f* ∈ *qbs-borel* $\rightarrow_Q$ $\mathbb{R}_Q$
and *uniform-f-qbs*[*qbs*]: *uniform-f* ∈ $\mathbb{R}_Q \bigotimes_Q \mathbb{R}_Q \rightarrow_Q \mathbb{R}_Q$
⟨*proof*⟩

lemma *uniform-f-measurable*[*measurable*]: *uniform-f* $\in$ *borel-measurable borel*
⟨*proof*⟩

lemma *nc-uniform-f*: *qbs-l* (*density-qbs lborel-qbs uniform-f*) *UNIV* = *1*
⟨*proof*⟩

corollary *measure-qbs-l-dens-to-dist-uniform-f*: *measure* (*qbs-l* (*density-qbs lborel-qbs uniform-f*)) *UNIV* = *1*
⟨*proof*⟩

lemma *uniform-f*:
shows *uniform-f* $\in$ *qbs-borel* $\to_Q$ $\mathbb{R}_Q$
and $\bigwedge x$. *uniform-f x* $\geq$ *0*
and *qbs-l* (*density-qbs lborel-qbs uniform-f*) *UNIV* $\neq$ *0*
and *qbs-integrable lborel-qbs uniform-f*
⟨*proof*⟩

lemma *qbs-l-dens-to-dist-uniform-f*:*qbs-l* (*dens-to-dist uniform-f*) = *density lborel* (λx. *ennreal* (*uniform-f x*))
⟨*proof*⟩

lemma *dens-to-dist uniform-f* = *Uniform 0 1* $\bigotimes_{Qmes}$ *Uniform 0 1*
⟨*proof*⟩

5.2.9 If then else

definition *gt* :: (*real* $\Rightarrow$ *real*) $\Rightarrow$ *real* $\Rightarrow$ *bool qbs-measure* **where**
gt $\equiv$ ($\lambda f\ r$. *do* {
    x $\leftarrow$ *dens-to-dist* (*normal-f 0 1*);
    if f x > *r*
    then return-qbs $\mathbb{B}_Q$ *True*
    else return-qbs $\mathbb{B}_Q$ *False*
    })

declare *normal-f*(*1*)[*of 1 0*,*simplified*]

lemma *gt-qbs*[*qbs*]: *gt* $\in$ *qbs-space* (($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$) $\Rightarrow_Q$ $\mathbb{R}_Q \Rightarrow_Q$ *monadP-qbs* $\mathbb{B}_Q$)
⟨*proof*⟩

lemma
assumes [*qbs*]: *f* $\in$ $\mathbb{R}_Q \to_Q \mathbb{R}_Q$
shows $\mathcal{P}$(*b in gt f r*. *b* = *True*) = $\mathcal{P}$(*x in std-normal-distribution*. *f x* > *r*) (**is** *?P1* = *?P2*)
⟨*proof*⟩

Examples from Staton [5, Sect. 2.2].

5.2.10 Weekend

Example from Staton [5, Sect. 2.2.1].

This example is formalized in Coq by Affeldt et al. [1].

definition *weekend* :: *bool qbs-measure* **where**
weekend ≡ *do* {
  let x = qbs-bernoulli (2 / 7);
    f = (λx. let r = if x then 3 else 10 in pmf (poisson-pmf r) 4)
  in query x f
  }

lemma *weekend-qbs*[*qbs*]:*weekend* ∈ *qbs-space* (*monadM-qbs* $\mathbb{B}_Q$)
⟨*proof*⟩

lemma *weekend-nc*:
defines *N* ≡ *2 / 7 ∗ pmf (poisson-pmf 3) 4 + 5 / 7 ∗ pmf (poisson-pmf 10) 4*
shows *qbs-l (density-qbs (bernoulli-pmf (2/7)) (λx. (pmf (poisson-pmf (if x then 3 else 10)) 4))) UNIV = N*
⟨*proof*⟩

lemma *qbs-l-weekend*:
defines *N* ≡ *2 / 7 ∗ pmf (poisson-pmf 3) 4 + 5 / 7 ∗ pmf (poisson-pmf 10) 4*
shows *qbs-l weekend = qbs-l (density-qbs (qbs-bernoulli (2 / 7)) (λx. ennreal (let r = if x then 3 else 10 in r ^ 4 ∗ exp (− r) / (fact 4 ∗ N))))* (**is** *?lhs = ?rhs*)
⟨*proof*⟩

lemma
defines *N* ≡ *2 / 7 ∗ pmf (poisson-pmf 3) 4 + 5 / 7 ∗ pmf (poisson-pmf 10) 4*
shows $\mathcal{P}$(*b in weekend. b = True*) = *2 / 7 ∗ (3^4 ∗ exp (− 3)) / fact 4 ∗ 1 / N*
⟨*proof*⟩

5.2.11 Whattime

Example from Staton [5, Sect. 2.2.3]

f is given as a parameter.

definition *whattime* :: (*real* ⇒ *real*) ⇒ *real qbs-measure* **where**
whattime ≡ (λ*f*. *do* {
  let T = Uniform 0 24 in
  query T (λt. let r = f t in
    exponential-density r (1 / 60))
  })

lemma *whattime-qbs*[*qbs*]: *whattime* ∈ ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$) $\Rightarrow_Q$ *monadM-qbs* $\mathbb{R}_Q$
⟨*proof*⟩

lemma *qbs-l-whattime-sub*:
assumes [*qbs*]: $f \in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$
shows *qbs-l* (*density-qbs* (*Uniform 0 24*) (λx. *exponential-density* (*f x*) (*1* / *60*))) = *density lborel* (λx. *indicator* {*0*<..<*24*} *x* / *24* ∗ *exponential-density* (*f x*) (*1* / *60*))
⟨*proof*⟩

lemma
assumes [*qbs*]: $f \in \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$ **and** [*measurable*]: $U \in$ *sets borel*
and $\bigwedge r.\ f\ r \geq 0$
defines $N \equiv (\int t{\in}\{0{<}..{<}24\}.\ (f\ t * exp\ (-\ 1/\ 60 * f\ t))\ \partial lborel)$
defines $N' \equiv (\int^{+} t{\in}\{0{<}..{<}24\}.\ (f\ t * exp\ (-\ 1/\ 60 * f\ t))\ \partial lborel)$
assumes $N' \neq 0$ **and** $N' \neq \infty$
shows $\mathcal{P}(t\ in\ whattime\ f.\ t \in U) = (\int t{\in}\{0{<}..{<}24\}{\cap}U.\ (f\ t * exp\ (-\ 1/\ 60 * f\ t))\ \partial lborel)\ /\ N$
⟨*proof*⟩

5.2.12 Distributions on Functions

definition *a-times-x* :: (*real* ⇒ *real*) *qbs-measure* **where**
a-times-x ≡ *do* {
    a ← *Uniform* (−*2*) *2*;
    return-qbs ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$) ($\lambda x$. *a* ∗ *x*)
  }

lemma *a-times-x-qbs*[*qbs*]: *a-times-x* ∈ *monadM-qbs* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$)
⟨*proof*⟩

lemma *a-times-x-qbsP*: *a-times-x* ∈ *monadP-qbs* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$)
⟨*proof*⟩

definition *a-times-x′* :: (*real* ⇒ *real*) *qbs-measure* **where**
a-times-x′ ≡ *do* {
    condition a-times-x (λf. *f 1* ≥ *0*)
  }

lemma *a-times-x′-qbs*[*qbs*]: *a-times-x′* ∈ *monadM-qbs* ($\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q$)
⟨*proof*⟩

lemma *prob-a-times-x*:
assumes [*measurable*]: *Measurable.pred borel P*
shows $\mathcal{P}(f\ in\ a\text{-}times\text{-}x.\ P\ (f\ r)) = \mathcal{P}(a\ in\ Uniform\ (-2)\ 2.\ P\ (a * r))$ (**is** *?lhs* = *?rhs*)
⟨*proof*⟩

lemma $\mathcal{P}(f\ in\ a\text{-}times\text{-}x'.\ f\ 1 \geq 1) = 1\ /\ 2$ (**is** *?P* = -)
⟨*proof*⟩

Almost everywhere, integrable, and integrations are also interpreted as programs.

lemma $(\lambda g\ f\ x.\ \mathit{if}\ (AE_Q\ y\ \mathit{in}\ g\ x.\ f\ x\ y \neq \infty)\ \mathit{then}\ (\int^+{}_Q\ y.\ f\ x\ y\ \partial(g\ x))\ \mathit{else}\ 0)$
$\in (\mathbb{R}_Q \Rightarrow_Q \mathit{monadM\text{-}qbs}\ \mathbb{R}_Q) \Rightarrow_Q (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_{Q\geq 0}) \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_{Q\geq 0}$
$\langle \mathit{proof} \rangle$

lemma $(\lambda g\ f\ x.\ \mathit{if}\ \mathit{qbs\text{-}integrable}\ (g\ x)\ (f\ x)\ \mathit{then}\ \mathit{Some}\ (\int_Q\ y.\ f\ x\ y\ \partial(g\ x))\ \mathit{else}\ \mathit{None})$
$\in (\mathbb{R}_Q \Rightarrow_Q \mathit{monadM\text{-}qbs}\ \mathbb{R}_Q) \Rightarrow_Q (\mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q \mathbb{R}_Q) \Rightarrow_Q \mathbb{R}_Q \Rightarrow_Q \mathit{option\text{-}qbs}\ \mathbb{R}_Q$
$\langle \mathit{proof} \rangle$

end

References

[1] R. Affeldt, C. Cohen, and A. Saito. Semantics of probabilistic programs using s-finite kernels in Coq. In *Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs*, CPP 2023, page 316, New York, NY, USA, 2023. Association for Computing Machinery.

[2] A. Sampson. Probabilistic programming. http://adriansampson.net/doc/ppl.html. Accessed: January 25. 2023.

[3] T. Sato, A. Aguirre, G. Barthe, M. Gaboardi, D. Garg, and J. Hsu. Formal verification of higher-order probabilistic programs: reasoning about approximation, convergence, bayesian inference, and optimization. *Proceedings of the ACM on Programming Languages*, 3(POPL):130, Jan 2019.

[4] S. Staton. Commutative semantics for probabilistic programming. In H. Yang, editor, *Programming Languages and Systems*, pages 855–879, Berlin, Heidelberg, 2017. Springer Berlin Heidelberg.

[5] S. Staton. *Probabilistic Programs as Measures*, page 4374. Cambridge University Press, 2020.

[6] H. Yang. Semantics of higher-order probabilistic programs with continuous distributions. https://alfa.di.uminho.pt/~nevrenato/probprogschool_slides/Hongseok.pdf. Accessed: February 8. 2023.